Oracle® Reference Architecture

Information Management

Release 3.1

E26368-03

July 2013

ORACLE®

ORA Information Management, Release 3.1

E26368-03

Primary Author: Stephen G. Bennett

Contributing Author: Dave Chappelle

Contributor: Andrew Bond, Doug Cackett

Contents

3 Common Information Management Standards

6 Logical View

List of Figures

List of Tables

Send Us Your Comments

ORA Information Management, Release 3.1

E26368-03

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this document?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number. You can send comments to us at its_feedback_ww_grp@oracle.com.

Preface

The Oracle Reference Architecture (ORA) is a product agnostic reference architecture based on architecture principles and best practices that are widely applicable. ORA can be implemented using a wide variety of products and technologies. ORA does not include any implementation artifacts for the prescribed architecture. Rather, ORA addresses the building of a modern, consistent IT architecture while minimizing the risk of product incompatibilities and obsolescence.

ORA is an extensible reference architecture that describes many facets of IT. It is comprised of several documents that cover core concepts of technology, along with other documents that build upon these core concepts to describe more complex technology strategies.

Developing robust business solutions requires a feature-rich and sophisticated information management platform that provides the capabilities necessary to organize, persist, manage, and retrieve enterprise information. This includes the capabilities to ensure accuracy, integrity, and consistency of the information as well as integration capabilities.

Technology is growing fast and getting increasingly complex, resulting in organizations struggling to find skills to develop solutions using the new tools and approaches. In order to overcome this challenge, organizations need a unified information management environment where end-to-end solutions are developed rapidly and reliably.

Document Purpose

This document covers the information management aspects of the Oracle Reference Architecture and describes important concepts, capabilities, principles, technologies, and several architecture views including conceptual, logical, product mapping, and deployment views that help frame the reference architecture.

This document describes a reference architecture for information management that can serve as a foundation for other business strategies, such as business analytics, business process management, or service-oriented architecture. In order to avoid overextending its scope, it does not attempt to describe how the architecture is applied to these other strategies. Instead, refer to the appropriate IT Strategies from Oracle (ITSO) documents. For example, refer to the *ITSO Business Analytics* documents which build on the concepts described in this document. Likewise, the security and management aspects of information management are covered by the *ORA Security* and *ORA Management and Monitoring* documents.

Audience

This document is primarily intended for enterprise architects, information architects, or others in a technology leadership position, to aid in reference architecture design. solution architects, information specialists, and business analysts may also find it useful and informative.

The material is designed for a technical audience that is interested in learning about the intricacies of information management and how infrastructure can be leveraged to satisfy an enterprise's information management needs. In-depth knowledge or specific expertise in information management fundamentals is not required.

Document Structure

This document is organized into chapters that build upon each other to form the basis of an information management reference architecture. The chapters are organized as follows:

Chapter 1, "Introduction" - introduces the subject of information management and describes the scope of this material.

Chapter 2, "Concepts" - describes important foundational concepts pertaining to information management.

Chapter 3, "Common Information Management Standards" - introduces several technology standards that apply to information management.

Chapter 4, "Architecture Principles" - offers a set of architecture principles that factor into the formation of the reference architecture.

Chapter 5, "Conceptual View" - defines a conceptual view based on a desired set of information management capabilities.

Chapter 6, "Logical View" - describes the logical view of the architecture.

Chapter 7, "Product Mapping View" - introduces Oracle products that apply to the architecture and maps them to the logical view.

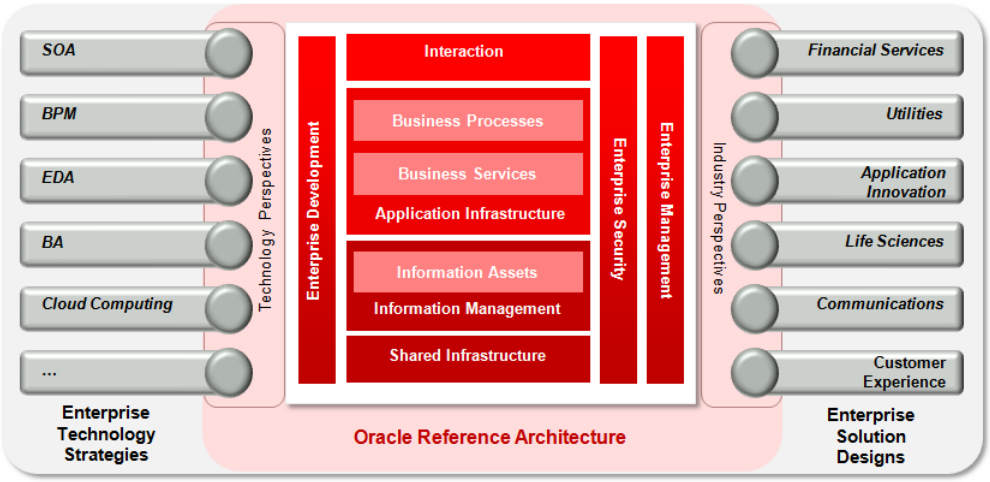
Chapter 8, "Deployment View" - describes an example deployment view of the architecture.

How to Use This Document

This document is designed to be read from beginning to end. However, each section is relatively self contained and could be read independently from the other sections. Persons familiar with information management concepts may wish to skip ahead to Chapter 5, "Conceptual View".

Introduction to IT Strategies from Oracle (ITSO)

IT Strategies from Oracle (ITSO) is a series of documentation and supporting material designed to enable organizations to develop an architecture-centric approach to enterprise-class IT initiatives. ITSO presents successful technology strategies and solution designs by defining universally adopted architecture concepts, principles, guidelines, standards, and patterns.



ITSO is made up of three primary elements:

- **Oracle Reference Architecture (ORA)** defines a detailed and consistent architecture for developing and integrating solutions based on Oracle technologies. The reference architecture offers architecture principles and guidance based on recommendations from technical experts across Oracle. It covers a broad spectrum of concerns pertaining to technology architecture, including middleware, database, hardware, processes, and services.
- **Enterprise Technology Strategies (ETS)** offer valuable guidance on the adoption of horizontal technologies for the enterprise. They explain how to successfully execute on a strategy by addressing concerns pertaining to architecture, technology, engineering, strategy, and governance. An organization can use this material to measure their maturity, develop their strategy, and achieve greater levels of adoption and success. In addition, each ETS extends the Oracle Reference Architecture by adding the unique capabilities and components provided by that particular technology. It offers a horizontal technology-based perspective of ORA.
- **Enterprise Solution Designs (ESD)** are cross-industry (applicable to many industries) and industry-specific (focused on a single vertical industry) solution perspectives based on the Oracle Reference Architecture. They adhere to the ORA principles and also draw on the best practices and guidelines provided in ETS collateral. They define the high level business processes, business functions, and software capabilities that are required to build enterprise-wide industry solutions. ESDs map the relevant application and technology products against solutions to illustrate how capabilities in Oracle's complete integrated stack can best meet the business, technical, and quality-of-service requirements.

ORA Information Management is one of the series of documents that comprise Oracle Reference Architecture. ORA Information Management describes important aspects of the Information Management layer that provides the capabilities necessary to acquire, organize, persist, secure, manage, and retrieve enterprise information. This includes capabilities to ensure accuracy, integrity, and consistency of the information as well as data integration capabilities.

Please consult the ITSO web site for a complete listing of ORA documents as well as other materials in the ITSO series.

Conventions

The following typeface conventions are used in this document:

Convention	Meaning
boldface text	Boldface type in text indicates a term defined in the text, the ***ITSO Master Glossary***, or in both locations.
italic text	Italics type in text indicates the name of a document or external reference.
underline text	Underline text indicates a hypertext link.

1
Introduction

Information is key to enabling competitive advantage and stakeholder value through information technology. However, to achieve this status, it must be delivered pervasively within the business as well as to the wider community and customer base, together with the required levels of management excellence to complete the transformation.

Most enterprise's operational systems have been purchased, built, leased, installed, and utilized over an extended period of time. Each of these operational systems has their own data needs and reporting functionality. Initially these needs are met within the boundary of the operational system. But over time, challenges arise as the needs of the business users expand beyond the boundary of the operational system. Inevitably, enterprises go down the easier route of copying data from one system to another or performing ad-hoc information integration to address these wider information access and reporting requirements.

Currently, IT groups spend too much time duplicating data, performing ad-hoc information integration, and data re-engineering; and business analysts spend too much of their time collecting and preparing data rather than analyzing it and taking action based on results.

What eventually emerges is a whole series of direct and indirect linkages which culminate in a multitude of databases, spreadsheets, and technologies that are very light on consistency and high in complexity. Traditionally, IT groups have tried to resolve this consistency and complexity through "stop-gap" approaches such as to replicate and/or consolidated operational data stores. But the information remains inconsistent, untrustworthy, and is unlikely to be used by business users for decision making with any degree of confidence.

If enterprises are to truly deliver information pervasively throughout a business and beyond, then an information management platform is required which must be capable of loading, transforming, integrating, and querying data in near real-time with the same level of accuracy and high availability as it is expected from every other operational system. With ever growing data volumes and deeper analysis requirements, it also follows that an information management platform must be able to scale out to meet future demands and manage the information lifecycle to reduce costs in a realistic and secure platform.

In addition to "managing information", enterprises are adopting Enterprise Technology Strategies (ETS) to support business efficiency and agility, such as Business Analytics (BA), Service Oriented Architecture (SOA), Business Process Management (BPM), and Event Driven Architecture (EDA). This has also raised the need to have a holistic approach to managing and delivering quality information in a timely fashion.

Information Management (IM) comprises of all the disciplines, architectures, policies, practices, and procedures that are concerned with the organization and coordination of

information including the acquisition techniques, containment strategies, retrieval mechanisms, and general lifecycle of data.

This ORA Information Management document offers a framework for an information management architecture that supports an overall information management strategy. This includes the capabilities to ensure accuracy, integrity, security, and consistency of the information as well as integration capabilities.

The remainder of this chapter introduces and provides a background into the key drivers pushing enterprises into evolving their treatment of data within an information management framework.

1.1 Data are Everywhere, Growing, and Picking up Speed

It is common knowledge that data volumes around structured and un/semi-structured data are growing at an ever increasing rate. Not only are data volumes growing, but also the number of sources, variety, velocity,and format of data; whether these are from an ever increasing internal/external application portfolio, data generating devices, or from syndicated data.

This increase in volume is due in part to organizations collecting vast amounts of real-time data from a variety of sources, from online social media data to highly granular transactional data to data from embedded sensors.

In addition, this increase in volume is due to limitations of an organizations information management strategy and governance program. Siloed application data, information management architecture limitations, mergers and acquisitions, and uncontrolled data replication, are leading to data proliferation leaving organizations with fragmented data silos that lack any real quality on which to base operational and analytical business decisions.

At the same time that data volumes are growing, so is the importance in turning it into useful information. But with ever increasing data silos being formed it is imperative that an information management platform can handle the ability to interoperate, consolidate, aggregate, and synchronize, with existing and also future unknown IT operational systems.

Convergence of all data into a single database is often not possible, either for technical, business, or logistical reasons. Regardless, information consumers should not be presented with duplicate and sometimes inconsistent data to navigate and make sense of.

Enterprises need an information management platform that can provide scalability, availability, and manageability, now and over time.

1.2 Data are Untrustworthy

Driven by investments in packaged applications and business analytical applications, users require trusted information for improved, actionable business insight. But uncontrolled data proliferation predictably leads to key enterprise processes being based on fragmented, duplicated, stale, incomplete, and conflicted data, leading to poor business insight.

In addition, challenges in meeting compliance demands, coupled with unexplained reporting inconsistencies, inevitably leads to loss of credibility in systems, and to users mistrusting the data.

Users have to access multiple siloed systems to access the data they require. This data typically have a multitude of definitions for the same business entity, and differing

levels of quality. Users then have the unenviable task of performing complex and difficult to resolve reconciliations over disagreements about whose data are correct.

Enterprises need an information management platform that improves the quality of data critical to making trusted business decisions and ensuring compliance.

1.3 Data are Inaccessible

There are numerous types of consumers of data, (such as customers, employees, partners, and suppliers), utilizing various delivery channels, who have varied data requirements. Invariably these stakeholders encounter challenges with accessing timely information to perform their job.

It is not uncommon for data to be unavailable, or for the mechanisms in which to access data to be overly complicated. When users do get access to the data it can sometimes be out of date, in the wrong format, or structured in a way that it is not easily understood. Consumers of data need the ability to access all the data they require, regardless of its format or where the data is stored but in keeping with their access privileges.

High-velocity data can bring high value, especially to volatile business processes. However, some of this data loses its operational value in a short time frame and therefore large volumes of dynamically changing data needs to be processed to extract the maximum value from this perishable data.

In addition, as data volumes grow, consumers are complaining of data overload with very little context, which inhibits decision making. These consumers require the data to be available in a number of historical and/or consolidated views so they can analyze it quicker and gain better business insights faster.

Consumers of data are increasingly requiring immediate access to consolidated data. But current technical mechanisms in acquiring and consolidating data have traditionally been executed overnight.

Enterprises require an information management platform that allows the access and updating of data from all the different sources in a secure, reliable, timely, uniform, and consistent manner, where complexity has been minimized and abstracted away.

1.4 Data are Inhibiting IT Productivity

With no reference architecture and information management platform in place, development teams have to go to great lengths to access the data they require, transform the data, sometimes even making additional copies of the data to solve technical and political issues. This can be compounded when the development teams do not fully understand the data they already have access to. In addition, legacy point-to-point data integration techniques are overly expensive, complex, brittle, inflexible, and hard to maintain and troubleshoot.

Enterprises require an information management platform that can address these integration and access challenges. Developers must also be allowed to focus more on "what" information to get and use rather than on "how" to get the data. Developers require a standards-based approach on which to access integrated views of information from various sources rather than using individual APIs particular to each data source. Developers and architects need a platform that allows a model-driven approach to handle information access and updates rather than always resorting to complex coding.

2

Concepts

This chapter describes some of the key concepts that pertain to information management.

2.1 What are Data, Information, and Knowledge

It is not uncommon for "Information" and "Data" to be spoken off interchangeably. What is the distinction and is it important to understand the distinction?

2.1.1 Data

Data is the representation of facts (e.g., text, numbers, images, sound, and video) for a particular subject and can be used as a basis for discussion, reasoning, measuring, and calculation. When it comes to IT, data tends to have its context and relationships held within a single IT system, and therefore can be seen as raw or unorganized. Data can also be seen as the building blocks for further analysis and investigation beyond a particular subject. The stewards for data are likely characterized as 'data producers'.

2.1.2 Information

Information is quality data that has been organized within a context of increased understanding and timely access. Peter Drucker stated that "Information is data endowed with relevance and purpose." This relevance and purpose is achieved by subjecting data from one or more sources to processes such as acquisition, enrichment, aggregation, filtering, and analysis, to add value and meaning within a certain context. Therefore information must have meaning to the business in a language and format that is understood, easily consumed, and is accessible in a timely manner. The stewards for information are likely to be not only producers but also consumers 'consumers' (i.e. users of the data being processed, aggregated or summarized).

2.1.3 Knowledge

Knowledge is the insight and intelligence achieved from the formal and informal experience such as intuitive exploration and consumption of information. Knowledge can be seen as actionable information as it enables stakeholders to act in an informed manner when they fully understand its context (e.g., via interpretation of patterns, trends, and comparisons).

2.2 Data Formats

To cater for a variety of data sources such as text documents, images, XML documents, and log files, data can be classified into three data formats - structured, semi-structured and unstructured.

Unfortunately these terms can be imprecise as their characteristics around storage, interpretation, and retrieval mechanisms start to blur; and in most cases structure, while not formally defined, can still be implied. Therefore, these terms should be seen as part of a data format spectrum rather than finite definitions. In addition, the classification should be based on how the data is structured and not where it is stored.

2.2.1 Structured

Structured data contains a number of elements grouped together (i.e. tabular or hierarchal) that has a structure that is highly identifiable. For self-describing data, there is no separation between the data and schema (e.g. XML Document), but the most common structured data has a separation between data and schema (e.g. relational table) with each row representing a new instance of the entity and the columns (or attributes) representing a set of data values of a particular type . This makes structured data easily readable and processed by systems as the data is organized into a highly identifiable structure.

2.2.2 Unstructured

Conversely unstructured data has no discernible identifiable structure and is not consistent from one record to the next. When unstructured data is text heavy it is more than likely interpretable by a human. But unstructured data can also include multimedia objects such as audio and video. The lack of a fixed structure makes interpreting and processing of the data via systems more challenging compared to data in a structured format.

2.2.3 Semi-Structured

As the name implies, semi-structured data can contain aspects of both structured and unstructured data. For instance, semi-structured data can contain a number of elements grouped together as in structured data, but each record can vary in the number of fields. Another example of semi-structured data is data that is large textually but has some attached contextual properties (e.g. metadata tags) that add value and aid in interpretation. Semi-structured data has the advantage of not being constrained by its schema as it can be easily being extended to cater for new data sources.

2.3 Data Domains

Data can be classified into various domains which defines its objectives and purpose. Below are example data domains.

2.3.1 Operational Data Domain

A company's operations are supported by applications that automate key business processes. These include areas such as sales, service, order management, manufacturing, purchasing, billing, accounts receivable, and accounts payable. These applications require significant amounts of data to function correctly. This includes data about the objects that are involved in operational transactions, as well as the operational data itself. For example, when a customer buys a product, the transaction

is managed by a sales application. The subjects of the transaction are the customer and the product. The operational data are the time, place, price, discount, payment methods, etc., used at the point of sale.

Operational data tends to be structured data found within relational databases or file systems that support business processes that are optimized for transactional purposes, (i.e. a large number of inserts, updates, and deletes to the data).

2.3.2 Content Domain

The amount of content data that companies have access to has grown dramatically in recent years. Content is primarily made up of a collection of unstructured/semi-unstructured user generated data including business documents (e.g. PDFs, spreadsheets), multi-media (i.e. images, audio, video), and web/social media (e.g. blogs, email, Twitter, Facebook).

To leverage the full value of this data, and to explore business opportunities, companies have to utilize this content not just in isolation, but also in conjunction with transactional data. This requires that content be indexed, searchable, and can be mined and analyzed for information.

2.3.3 System Data Domain

System data is data that is primarily generated by machine (aka machine-generated). This domain also encapsulates data regarding the monitoring of user activity. In effect, system data can be seen as "point in time" data. There is a vast amount of system data available today that is being utilized. Log files from web servers are utilized to produce web site analytics, and application logs are utilized to assist with root cause analysis.

Machine-generated data is an area where there will be substantial growth in the coming years. When taking into consideration initiatives such as the "The Internet of Things", where sensors are embedded into physical objects and gain the ability to communicate, vastly more data will be generated. There is a vast number of platforms and devices that generate sensor and log data, including network/telecom equipment, smart phones and tablets, and increasingly more equipment such as smart meters, medical devices, cars, trains, planes, heating systems, refrigeration units, RFID (Radio Frequency Identification) tags, etc.

Due to their sources (e.g. computers, devices), system data are invariably large in volume and can be produced in high velocity in a short period of time; and as they tend to be "Data of Observation" they tend to be un/semi-structured and not updateable. The ability to store, process and analyze this data offers great potential to companies (e.g. to streamline current operations in controlling costs and increasing quality).

2.3.4 Authoritative Data Domain

Authoritative data are the highest quality data found within an enterprise, and they give operational data context. There are two classifications of authoritative data: master data and reference data.

2.3.4.1 Master Data

Master data represents the standardized key business entities that support operational data such as customer, product, financial structures, and sites. In addition to supporting operational data, master data also supports analytical data by representing key dimensions on which analytics is accomplished. Master data are in effect the

trusted, quality, single version of the truth for key business entities that an enterprise requires. As master data supports operational applications, it also must support high volume transaction rates.

2.3.4.2 Reference Data

Reference Data represents internally and externally sourced, standardized entities that apply a classification context to other data domains, (e.g. currency codes, status codes). Lower data volumes, fewer changes, and not as business critical as master data tend to lead organizations in not enforcing the same level of data quality rigor to reference data.

It is common for reference data to utilize a data caching approach so that applications have access to it without the overhead of having to continually access a relational database. With reference data changing slowly, this is a good approach. But when there is an update to the reference data it will need to be synchronized across all data caches.

2.3.5 Analytical Data Domain

Analytical data utilizes operational, content, and system data to enable enterprises to discover new ways to strategize, plan, optimize business operations, and capture new market opportunities. This data supports the ad-hoc queries, exploration, business insight, predictive analysis, and planning needs of an enterprise. For instance:

- Customer buying patterns are analyzed to identify churn, profitability, and marketing segmentation.
- Suppliers are categorized, based on performance characteristics over time, for better supply chain decisions.
- Product behavior is scrutinized over long periods to identify failure patterns.

To support these needs, data are stored and organized to enhance mining, aggregation, and exploration. A portion of the data may be de-normalized, distributed, and/or optimized for analytical processing and querying.

2.3.6 Metadata Domain

Metadata is a broad term that articulates several business and IT aspects of an enterprises data, IT systems, and business operational environment. It is data that is used to describe other data. Metadata provides context and purpose for data around definition, relationship, creation, policy, search, retrieval, transformation, and lineage - too name just a few. Metadata has become very critical in both the management and operation of information management.

2.4 Data Classification

Not all data are equal. Some are freely available in the public domain, while others have varying degrees of sensitivity. Some are frequently used, while some are rarely used. Some have great importance to the company, while others have little value over the long term.

An important aspect of information management is determining an appropriate classification scheme(s) which assists in determining the appropriate architecture requirements.

Refer to the *ORA Security* document for example data security classifications.

2.5 Data Persistance

In general, organizations today utilize relational databases and file systems to address the majority of their data persistence needs. Even though other data persistence mechanisms have been available (e.g. hierarchical, network, object), relational is by far the most prevalent.

But more recently, requirements to support increasing dataset complexity, performance and scalability challenges, are making organizations adopt additional data persistence mechanisms. This enables an organization to utilize the appropriate persistence mechanisms for each use case, which has come to be known as polyglot persistence.

The following sections describe several traditional, and some more recently favored data persistent mechanisms. It is important to understand the characteristics and the potential usage of the data to evaluate to evaluate which persistence mechanisms to use.

2.5.1 Relational Database

The relational database has become the de-facto standard in data persistence due to its ability to support a very broad class of uses and its use of a rich query language (i.e. SQL). Relational databases are mature, well understood, heavily supported by frameworks and tools, and can support complex data relationships.

A relational database allows the definition of two-dimensional data structures (tables), storage and retrieval operations, and integrity constraints. A table contains a collection of records (a row), and each row contains the same fields (columns). Columns may have indexes to increase the speed of queries.

The relational database has demonstrated its flexibility with its ease to add additional tables and relational paths at design-time and its ability to utilize these relational paths (joins) at retrieval time.

2.5.2 Disk File System

A disk file system utilizes disk storage, whereby files are named and placed logically for storage and retrieval. A disk file system manages a folder/directory structure, which provides an index to the files, and it defines the syntax used to access them. There are numerous file systems in use (e.g. UFS, ext3, NTFS, and ZFS).

2.5.3 Multi-Dimensional

Multidimensional utilizes cube structures to organize data and its relationships, where each cell contains aggregated data related to each of its dimensions. Its high performance has made it the most popular database structure when it comes to enabling online analytical processing (OLAP).

2.5.4 In-Memory Database

In contrast to relational databases, which employ a disk storage mechanism, an in-memory database (IMDB) is a database system that primarily relies on physical memory for data storage. Accessing data in memory reduces I/O when querying the data which provides for faster and more predictable performance than disk. This is obviously advantageous for applications where fast and predictable response time is critical.

IMDB's come in many shapes and sizes, but capabilities such as interoperability, recoverability, scalability, security, and availability are imperative if IMDB's are not to

be seen as custom-built memory structures such as hash tables. For instance, a relational IMDB can be accessed with standard ODBC and JDBC, providing the rich functionality of the SQL query language, while utilizing disk persistence for recoverability.

2.5.5 In-Memory Data Grid

In-memory data grids (IMDG) provide fast access to frequently used data by caching data and avoiding expensive requests to back-end data sources. This decoupling of consumers from the physical data sources compensates for the relatively slow performance of disk drives by keeping objects in memory.

IMDGs manage data objects in memory across many servers by automatically and dynamically partitioning data while ensuring continuous data availability and transactional integrity. This increases performance by moving data closer to the consumers for efficient access.

IMDGs are evolving beyond distributed caching and can provide a rich data processing model so processing can be farmed out to where the data is, and results returned back to the client.

2.5.6 Distributed File Systems

Distributed file systems (DFS) are similar to native files systems (e.g. ext3) in that they utilize disk storage for the storage and retrieval of files. Distributed file systems support a traditional hierarchical file organization in which a user or an application can create directories and store files inside them. But instead of placing files on a single storage unit, distributed file systems partition and distribute data across a number of interconnected machines. This enables distributed file systems to handle very large datasets (e.g. petabytes), such as can be encountered when capturing system data. Even though the data is distributed over many machines, distributed file systems abstract away the complexity and make it is possible to access the data files as one seamless file system.

There are numerous distributed file systems in use but the most well-known is the Hadoop Distributed File System (HDFS). HDFS sits on top of a native file system such as ext3, and connects machines (e.g. nodes) together into a cluster, over which data is distributed across all nodes at load time. Using hashing, HDFS splits files into sequenced blocks of pre-defined sizes, which are stored as standard files on the relevant data nodes in a set of directories. Therefore different blocks from the same file will be stored on different nodes. HDFS also provides a level of fault tolerance for massive amounts of data by automatically maintaining multiple copies of data by replicating blocks across multiple nodes.

2.5.7 NoSQL

NoSQL (Not only SQL) is a "broad" class of special purpose non-relational database systems; in truth this class of database systems should probably be named "Not only Relational". With no prevailing standards and no precise definition of what constitutes a NoSQL database, a large number of products have been branded as NoSQL implementations. These products have been implemented differently and have different features, performance, and scalability characteristics.

NoSQL databases tend not to have a pre-defined schema thus allowing for flexible data models that can evolve without forcing the existing data to be restructured. For example, records can have a variable number of fields that can vary from record to record. NoSQL implementations tend to rely on applications to understand the record

contents and enforce any required integrity. Therefore, NoSQL databases are developer-centric as they develop and manage the schema rather than the more traditional DBA-centric where the DBA manages the schema.

A large push for NoSQL databases has been their ability to scale horizontally (scale out) to cater for increasing data volumes. NoSQL databases tend to utilize a distributed file system so that data is partitioned into shards and distributed to other nodes. To increase capacity, additional nodes are made available and in some databases the shards are then automatically re-partitioned.

To enable a level of fault tolerance, these shards can be automatically replicated asynchronously to other nodes. One side effect of this strategy is that data is not immediately replicated and could be lost if a failure occurs prior to replication.

NoSQL databases prioritize performance and availability over other database system properties such as consistency, and thus either do not support or relax the implementation of ACID (Atomicity, Consistency, Isolation, and Durability) capabilities. Many NoSQL databases utilize an eventual consistency model for any database updates. In effect, this means that updates made at one replica will be transmitted asynchronously to the others, making NoSQL databases more aligned with BASE (Basically Available, Software State, and Eventually Consistent) capabilities. Some NoSQL databases utilize an optimistic concurrency model, while even fewer offer the ability to enable full transactional support (But not two-phase commit).

Generally, the best places to use NoSQL technology is where fast fetching of relatively simple data sets matters most, where data structure flexibility is important and strict data consistency is not required. Below are four broad classifications of NoSQL databases.

2.5.7.1 Key-Value

Key-Value databases perform operations on simple records comprised of a key and a value. It is basically a two column table containing a key and a value. The keys are typically stored and searched using a standard B-tree structure, while the value field has the flexibility to store almost any data structure. This means that applications must understand the structure of data in the value field.

Key-Value databases have minimal indexing and make available basic APIs such as get, put, and delete. Compared to more relational databases, they are very limited because they tend to provide only a single way to efficiently access values. These databases are designed for applications where simple primary key lookups and key-value record writes are their primary requirement. Therefore, applications that need quick single-record or sub-record searches often fit this model very well.

Therefore, key-value databases can be used as collections, dictionaries, and caches. Example uses include sensor data capture, user profile management, personalization attributes, shopping cart contents, and product lists.

2.5.7.2 Columnar

Columnar databases are designed for read, write, and update on groups of columns and records. Records are divided into column families, which are typically co-located on disk. Columnar databases store column values for multiple rows in the same block instead of storing complete rows of multiple columns within the same block. This can be faster when queries typically require fewer columns and more rows. So, rather than fetch lots of blocks and throwing away some of the data, columnar databases only perform I/Os to the blocks corresponding to columns that are actually being read or updated.

Basically, columnar databases are designed for applications where reading a group of records and accessing a small subset of columns is their primary requirement. Applications that need to do aggregation of data often fit this model because they can read multiple records at a time with a single I/O. But columnar databases tend to be slower for random I/O, small retrievals, and inserts.

2.5.7.3 Document

Document databases focus on the storage and access of document-oriented, self-describing data structures, (e.g. XML, JSON, BSON), as opposed to rows or records. The data structure of each document can be unique. Document databases often support multi indexing and nested documents. Unlike Key-Value databases, document databases allow queries beyond the key field. Document databases allow queries on the entire data structure. In addition, documents themselves can be addressed by unique URLs.

Document databases are best utilized for applications that are XML- or JSON/BSON-centric, view their records as documents, and need to perform document-centric lookups. Because document databases can support a more complex data model than Key-Value databases, they are good at handling use cases such as content management, session management, and profile data.

2.5.7.4 Graph

Graph databases store and manage objects and the relationships between objects. In graph databases these objects are called nodes and the relationships between them are called edges. The nodes and relationships can also contain an arbitrary amount of properties (key-value pairs) which enables a powerful way to represent data. The use of nodes, edges, and properties as fundamental elements in a graph database provides a sharp contrast to the tables, rows, and columns of a relational database.

As with document databases, graph databases may be implemented with different storage techniques. The graph nature of these databases exists mainly in the data model presented to the consuming application. Under the hood, a graph database may be a key/value database, a columnar database, or some combination of these and other data persistence mechanisms.

Graph databases possess the ability for a node to store a collection of IDs of other nodes. Navigating from one node to its related nodes can be performed quickly without maintaining secondary indexes. When multi-value collections are stored in both directions of an edge, the graph database can quickly navigate many-to-many relationships in either direction. Edges can be added or removed at any time, allowing one-to-many and many-to-many relationships to be expressed easily and avoiding relationship tables to accommodate the many-to-many join.

Graph databases are best suited where applications require the ability to rapidly traverse objects along their relationships, known as traversing the graph. Examples include social relationships, public transport links, road maps, and network topologies.

2.6 Data Partitioning

Data partitioning enables data to be subdivided into smaller pieces to enhance its overall performance, manageability, and availability.

2.6.1 Relational Data Partitioning

Relational data partitioning allows tables, indexes, and index-organized tables to be subdivided into smaller pieces called partitions. Each partition has its own name, is managed and accessed at a finer level of granularity, optionally has its own storage characteristics, while being entirely transparent to consuming applications.

Relational data partitioning may improve query performance and greatly reduces the total cost of ownership, by using a "tiered archiving" approach of keeping older relevant information still online on low cost storage devices.

2.6.2 Distributed Data Partitioning

To assist in the processing of large quantities of data (e.g. content and system data), many parallel data processing frameworks rely on utilizing distributed data partitioning capabilities such as those offered by distributed file systems and distributed NoSQL databases.

Distributed data partitioning utilizes algorithms (e.g. key hashing) and/or user defined distribution strategies to distribute data across a number of nodes (aka shards or data partitions). To increase capacity, additional nodes are made available and then the data partitions can then be rebalanced. To further increase performance, as well as the level of fault tolerance, the data partitions can be automatically replicated asynchronously to other nodes.

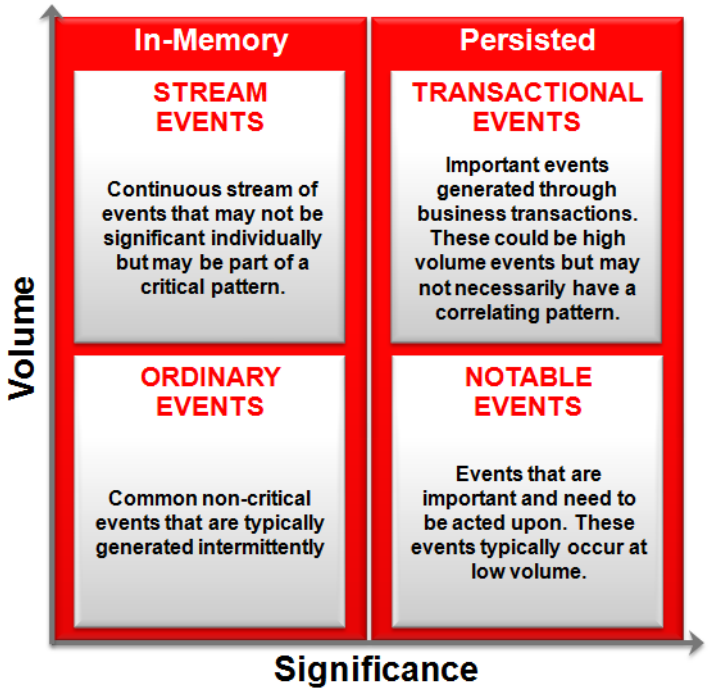
2.7 Data Streams / Events

Data streams represent data that is changing constantly, often exclusively through insertions of new elements called events. Many types of applications generate streams of events as opposed to data sets, including sensor data applications, financial tickers, network performance measuring tools, network monitoring and traffic management applications, and clickstream analysis tools.

An event is a change in any state that indicates a business opportunity, threat, anomaly, or just a notification. It is something that happens inside or outside the organization that should be noted or acted upon. Events may be business data centric or technology data related.

Technical events (or raw events) are fine grained, while business centric events may be defined at various levels of granularity. Another approach to classifying events is based on how they relate to volume and significance, see Figure 2–1, "Event Classifications".

Figure 2–1 Event Classifications



Highly significant events must be transported and processed reliably using reliable messaging and high availability techniques. Insignificant events are, in general, dispensable and can be implemented using in-memory messaging for speed and cost reasons.

For more details regarding events refer to the *ITSO EDA documents*.

2.8 Data Modeling

This section describes concepts related to various forms of data and information modeling and how they apply to information management.

2.8.1 Relationship Modeling

There are various levels of abstraction when developing relational data models.

2.8.1.1 Conceptual

Conceptual modeling is the process of defining and modeling the significant subject areas of the business and their relationships, independent of technology. This in turn starts to form the basis of the business glossary. Once validated and approved, the conceptual model can become the basis of a logical model.

2.8.1.2 Logical

A logical model continues from where the conceptual model ends, by modeling the business entities within a subject area, identifying attributes, relationships, and business rules. The final work product of the logical model is an entity relationship diagram (ERD). An ERD is a visual representation of data objects and their logical structure as entities, attributes, and relationships. An entity corresponds to a table, while relationships illustrate how two entities share information in the data structure.

The ERD also illustrates cardinality, which is how many instances of an entity relate to one instance of another entity. Once validated and approved, the logical data model can become the basis of a physical data model and inform the design of a database.

2.8.1.3 Physical

A physical model defines the physical means used to store data taking into consideration the capabilities of the database. Therefore the model is tightly coupled to the technology and includes table definitions, index definitions, and partitioning.

2.8.2 Dimensional Modeling

Dimensional modeling is a method of organizing data in a way that represents natural business hierarchies. For instance, a sales representative works in a sales district, which is part of a sales region, within a sales geography. Likewise, an item that is sold can belong to a product category, which in turn can belong to a higher level product group. Each of these two hierarchies is in a dimension.

For more details refer to the *ORA Business Analytics Foundation* document.

2.8.3 NoSQL Modeling

Relational modeling has a strong emphasis on normalization. Relational modeling practitioners normally normalize data models to the 3rd normal form, and only diverge from this for performance reasons. As previously stated, the relational database has become the de-facto standard in data persistence due to its ability to support a very broad class of uses and its flexibility to support various queries at run-time.

NoSQL databases tend not to support joins and therefore the joins have to be designed into each application. Therefore NoSQL modeling is dictated by an application's specific data access patterns. The downside of this situation is that a NoSQL database may lead to a tight coupling of the data and its data access methods within an application, unlike an application that utilizes a relational database where queries and indexes can be updated outside of the application code.

Some aspects of NoSQL modeling are to address the lack of availability of ACID properties within NoSQL databases. For instance, de-normalization can be utilized to simplify and optimize query processing by duplicating certain data fields rather than providing a lookup table. While this addresses the lack of join functionality in some NoSQL databases, it does lead to increased data volume and potential integrity challenges. This de-normalization can be taken either further by embedding nested objects with a single object (aka aggregation). This enables the ability to update multiple objects within one transaction as NoSQL databases tend to lack the functionality of multiple updates within a single transaction.

2.9 Data Processing

2.9.1 Online Transaction Processing

Online Transaction Processing (OLTP) is an approach to support transaction oriented applications for efficient real time querying and transaction processing.

The transactional data are stored in On-Line Transaction Processing (OLTP) tables that are designed to support high volume low latency access and update.

Operational data systems have been called OLTP (Online Transactional Processing System). An OLTP system has current transactions and is very write intensive. An OLTP system has a large number of inserts, updates, and deletes to the data.

Examples of OLTP systems are order entry, retail sales, and financial transaction systems.

2.9.2 Online Analytical Processing

In contrast to OLTP, Online Analytical Processing (OLAP) is an approach to provide business analytics by using dimensional models to satisfy multi-dimensional queries. The output of a query is often represented in a matrix or pivot table form.

Multi-dimensional queries define the subset of dimensional data to be analyzed. They define the dimensions, hierarchies and levels from which results sets are returned. For example, a query might ask for all sales data from store 123 pertaining to brand X for the previous six months. The query might also ask to compare these results with the same data from the previous year.

See Table 2–1, " OLTP vs. OLAP" for example characteristics to see the differences between OLAP and OLTP.

Table 2–1 OLTP vs. OLAP

OLTP	OLAP
Operational Data	Analytical Data
Write Intensive	Read Intensive
Current Data + Some Historical	Summarized Historical Data
Simple Queries	Complex Queries
Few Detailed Records	Large Volume of Records
Normalized	De-normalized or Partial
Volatile	Non-Volatile

Refer to *ORA Business Analytics Foundation* document for more details on OLAP and OLAP Cubes

2.9.3 Big Data Processing

Big Data is a term applied to data sets whose size, structure, and acquisition speed is beyond the capability of conventional software tools used for capturing, ingesting, managing, and processing data. While the processing of Big Data can be applied to various scenarios (e.g. document/image conversions), the primarily focus currently is on including untapped data to improve decision making.

Combined with the complexity of analysis and commercial imperative to create value from it, Big Data has led to a new generation of technologies, tools, and architectures.

Therefore, the term Big Data tends to be used in multiple ways, often referring to both the type of data being managed as well as the techniques and technology used to store, process, and analyze it. Big Data is still early in its maturation and therefore new processing engines, analytical tools, specialized hardware, data sources, techniques, and analytical needs will arise. Any architecture that intends to address Big Data Processing must be flexible enough to handle these enhancements. The term Big Data could very well fade in the future once these architectures, techniques, technologies,

and tools have matured and become common components of an overall information management architecture.

Big Data is often characterized by the 3V's: volume, velocity, and variety[1]. These characteristics all contribute to the growing need for larger and larger data storage and processing capabilities. But Oracle believes that another overarching Big Data characteristic must be included, and that is Value.

2.9.3.1 Volume

Volume refers to the amount of data that needs to be captured, processed and analyzed due to an ever-increasing number of systems and devices that are generating extremely large quantities of data. Examples such as RFID tags, GPS sensors, medical sensor devices, and click-stream data (website interactions) often generate terabytes, and in some cases petabytes, worth of data.

Rather than state in any absolute sense what volume of data should be regarded as Big Data, it is best to consider whether an organization's existing architecture, techniques, and tools can address these large data sets in a speedy and cost effective manner.

2.9.3.2 Velocity

Velocity refers to the speed and frequency at which data is being received and the need for it to be processed and analyzed. This data can often be streamed to a velocity where 100's of millions of events per day are generated and evaluated, (e.g. fraud detection).

In addition, one must consider time sensitive analytical requirements. They typically require data analysis and pattern identification while data is in motion, as opposed to loading data in a database for later analysis.

2.9.3.3 Variety

Variety refers to the increasing formats and structures of data being generated by a growing number of sources (e.g. social network interactions, smart phones, smart meters, and medical devices).

Traditionally, information management strategies primarily focused on structured, modeled data and the use of relational databases. This tended to lead companies to the wrong assumption: that unstructured data had little business value. Companies that did understand the importance of unstructured data as part of an overall information management strategy encountered complexity and cost challenges when utilizing conventional information management tools.

The Big Data ingestion and analytical needs faced by businesses are fundamentally very different from what traditional analytical processing addresses - which have been primarily historical, aggregate after-the-fact analysis with fixed schemas, data relationships and static visualizations. What is needed is a new way to capture, process, analyze, and visualize data characterized by the 4 V's.

Big Data architectures, techniques, and tools have made it possible to capture, process, and correlate an ever increasing variety of data that can now be included as part of an overall information management strategy.

2.9.3.4 Value

Value refers to finding business value in the storing and processing of new data sources, which is necessary to justify capturing and keeping it in the first place.

[1] 3-D Data Management: Controlling Data Volume, Velocity and Variety

Companies struggle to manage and exploit much of the data already at their disposal. Therefore companies will need to decide where additional investments are made.

For example, data within the financial industry can have a very low "value-to-noise ratio" (a.k.a. low information density) where more and more information needs to be inspected and filtered to get at the relevant information.

2.10 Information Stores

Information stores are data repositories that are utilized to support information provisioning. Some examples include an operational data store, data mart, and data warehouse.

2.10.1 Operational Data Store

An Operational Data Store (ODS) is a copy of one or more operational databases that is most often used for reporting purposes. The ODS is a subject-oriented, integrated, current, and volatile collection of data used to support the tactical decision-making process for the enterprise

The reason an ODS exists is generally twofold:

- The OLTP database can't handle the additional load placed on it by users running reports or doing ad hoc queries. A separate system with a copy of OLTP data is needed.
- A more universal reporting and analysis solution is not available to manage the data, so a tactical decision is made to deploy a solution for the requirements at hand.

An ODS usually contains incremental snapshots of OLTP data, and is used by a limited number of people. The data may have been cleansed or quarantined before being loaded into the ODS.

2.10.2 Data Mart

A Data Mart is a storehouse of data, generally collected from a number of systems that make up a business process. For example, finance, customer service, warehouse management, etc. Data marts can be thought of as data warehouses with a limited scope.

They are often created to satisfy the reporting and analysis needs of a particular department. Since data marts maintain a collection of data that are used for analysis purposes, they often follow dimensional modeling schemes.

2.10.3 Data Warehouse

A Data Warehouse is a specially prepared repository of data designed to support decision making by enabling reporting and analytics. The Data Warehouse can be seen as a primary storehouse of cleansed, transformed, and consolidated historical data for an organization that provides business users with a time-based, integrated view of cross-functional data.

To create the Data Warehouse, data are extracted from source systems (i.e. operational systems and external sources), cleansed, transformed and summarized, and loaded into the Data Warehouse.

2.11 Governance

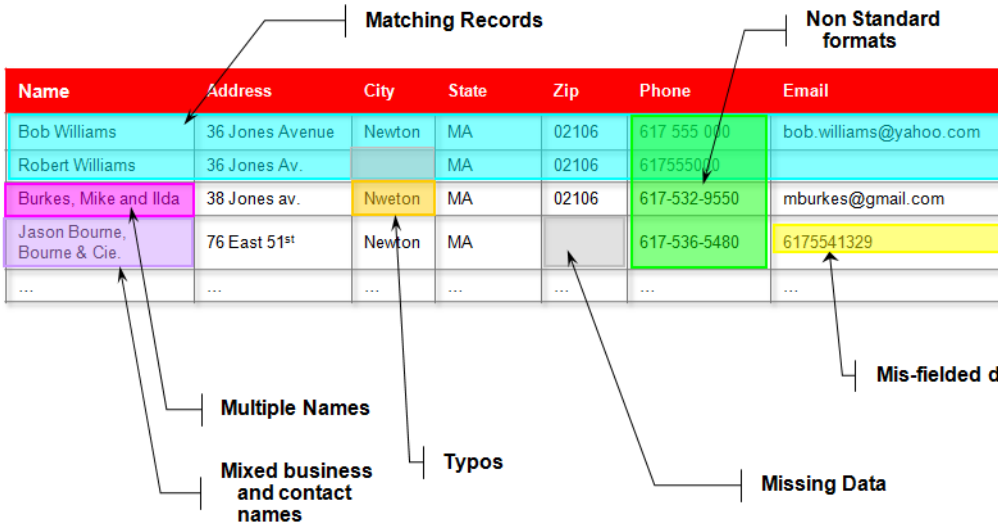
The Data Governance Institute states that governance is a system of decision rights and accountabilities for information-related processes, executed according to agreed-upon models which describe who can take what actions with what information, and when, under what circumstances, using what methods[2].

2.11.1 Data Quality

A critical function of governance is to ensure data quality. Data Quality Management is an approach in making and keeping data in a state of completeness, validity, consistency, timeliness, and accuracy, that makes it appropriate for a specific use. Data must be fit for purpose and accurately reflect the real world entities and facts for which they represent.

Many factors can adversely affect the quality of data, including incomplete or inaccurate data entry, inconsistent representation of data fields across solutions, and duplication of data within or across solutions. See Figure 2–2, "Example of Data Quality Issues".

Figure 2–2 Example of Data Quality Issues



In addition, normal business/IT events such as new products/services, mergers & acquisitions, and new applications, place more pressure on the quality of the data utilized by users, business process, services, and systems. Therefore, data quality requires being pervasive and continuous to make sure that enterprise data does not decay and lose value.

Obviously, it is best to address data quality issues at the source. If defects can't be prevented, the best place to clean the data is in the source system so the defects cannot spread to other downstream systems. If this is not possible in a timely manner, then to minimize costs data quality should be done as close as possible to the source data stores.

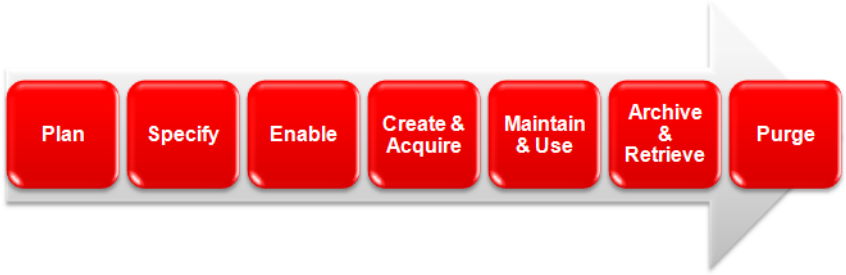
[2] Data Governance Institute - Dictionary Definition

2.11.2 Information Lifecycle Management

The Storage Networking Industry Association (SNIA) states that Information Lifecycle Management (ILM) comprises of the policies, processes, practices, and tools used to align the business value of information with the most appropriate and cost effective infrastructure from the time information is created through its final disposition. Information is aligned with business requirements through management policies and service levels associated with applications, metadata, and data[3].

Data must be treated as an enterprise strategic asset which must be governed and managed. Like any other asset it should be managed and governed in terms of a lifecycle. Figure 2–3, "Example Information Lifecycle" highlights an example lifecycle . No matter what information lifecycle is utilized, it is important that it is applicable regardless of data domain, data classification, and data format of the data in question.

Figure 2–3 Example Information Lifecycle



Traditionally, data has been classified on frequency of usage, and then the associated storage tiers were created for the various identified data usage classifications. During the lifetime of the data, it is then migrated between the storage tiers and access to it is controlled. Eventually the data may have been archived. Therefore as a rule, newer data, and data that were accessed more frequently, was stored on faster, but more expensive storage media, while less critical data was stored on cheaper, but slower media.

2.11.3 Governance for Big Data

With data quality, information privacy, and information lifecycle management, Big Data has compounded the challenges around information governance.

One train of thought is that information governance is not important within the context of Big Data. For instance, the inference is that the bigger the dataset the less impact the individual data quality errors will have on the overall data processing outcome. But information governance still matters when it comes to Big Data.

In the past, information governance primarily focused on structured data, but organizations should now consider defining and applying information governance policies for unstructured/semi-structured data. For example, addressing misspelled words and identifying synonyms within textual data can assist in filtering out noise and aid in entity identification and data classification for later analysis.

Due to Big Data's characteristics (e.g. Volume, Velocity, Value, and Variety) and the sharp increase of data sources, applying information governance (e.g. resolving duplications, missing values, and standardization) against Big Data can be challenging. Big Data Processing brings together structured and unstructured data,

[3] Storage Networking Industry Association (SNIA) - Dictonary Definition

data in motion, and data at rest, internal data, and external data. These new and disparate data sources may have little or no control over the quality of data being created and therefore may need to be standardized with the same set of rules, and then validated and monitored.

Within the realm of Big Data Processing, organizations have to accept the reality that data that is processed in a low latency fashion will be dirty. This is due to the shorten timeframes in which appropriate governance policies can be applied. Therefore, it is not always possible or preferred to apply all possible information governance policies against Big Data. Organizations should consider applying "Good Enough Governance" (GEG). Using GEG, organizations must establish and define the minimum information governance levels that are required before Big Data is useful for business purposes, taking into consideration the following characteristics:

- **Context/Potential Use** - Understand the context and potential uses of the data in question so that it is possible to apply different levels of information governance to different data types and data sources. For example, the quality of data within a twitter feed may not be as critical as the sensor data being produced by a medical device.
- **Risk/Reward** - Understand the level of risk versus reward that is acceptable to the business when focusing on the general indicator of the data rather than its exact precision.
- **Regulatory** - Understand the need to adhere to internal and external regulatory requirements. Consider the privacy concerns surrounding social media and sensor data (e.g. smart meters revealing empty households).
- **Information Lifecycle** - Recognize that the information lifecycle management aspects of Big Data do not rely solely on its age or how often it's accessed. Therefore a systematic policy-based approach is needed whereby Big Data is governed and managed through each phase of an established information lifecycle. Organizations need to understand how Big Data evolves, determine how it grows, monitor how its usage changes over time, and decide how long it should be retained and how it should be disposed, (e.g. the retention period of smart meter data), whilst adhering to all the rules and regulations that now apply. Without establishing archiving policies, storage costs could easily spiral out of control. Therefore, making sure you are keeping in line with regulatory requirements, it might make sense to not archive all machine-generated data but rather the data-set that was filtered/aggregated and derived.
- **Right Place** - Understand the appropriate place to apply information governance policies, as applying information governance policies at the origin of the data is not always possible or preferential (e.g. cleansing outliers may hide faulty medical devices that are transmitting invalid data). It may also be preferential to apply information governance policies (e.g. standardization and enrichment) once Big Data processing has been completed (e.g. outcome is being ingested into a downstream system such as a Data Warehouse).
- **Right Time** - With reduced time windows for analysis and action, it is important for organizations to understand the time available before the data is needed to make a business decision. Applying data quality techniques will invariably delay access to the cleansed data. For example, the transitory nature of streamed data limits the amount of information governance policies that can be applied before its time to value expires.

In essence, consider where the data came from, how the data will be processed, when the data is required, how the data will be consumed, and most importantly, what decisions will be made.

Governance of Big Data should not been seen as a siloed effort but as part of an overall enterprise Information Governance program. Initially, this includes extending the information governance charter to cater for Big Data, and identifying critical data elements that have the most impact on the business.

2.12 Business Intelligence & Business Analytics

Business intelligence (BI) is a general term that can be interpreted in many ways. Forrester Research defines it as "a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information. It allows business users to make informed business decisions with real-time data that can put a company ahead of its competitors[4]."

For many years BI was a term used to encompass all aspects of methodology, process, architecture, and technology that gathered and presented information to support business decisions. It encompassed the provisioning of information, (also known as data warehousing), generation of reports, presentation of dashboards, and analysis of historical data. BI also included forward-looking disciplines such as predictive modeling and simulation.

The recent emergence of "Big Data", and technologies that support the analysis of less structured data, has greatly promoted the role of advanced analytics. As a result, the term "Business Analytics" has become a hot topic in the industry.

Like BI, Business Analytics (BA) is defined in many ways. Thomas Davenport and Jeanne Harris define analytics as "the extensive use of data, statistical and quantitative analysis, explanatory and predictive models, and fact-based management to drive decisions and actions"[5]. This effectively positions BA and BI as peers, overlapping in some areas such as structured data analytics (e.g. OLAP), predictive modeling, and various applications of statistical analysis.

Lately BA has also become an umbrella term within the industry that encompasses the emerging aspects of analytics as well as the traditional aspects of BI. BI is positioned mainly as a descriptive form of analytics. It complements other forms such as predictive, prescriptive[6], and exploratory analytics[7]. The precise number and definition of forms depends on one's source of reference. What is fairly consistent is the encapsulation of BI within a multi-faceted definition of BA. Below are the high-level definition of the various forms of analytics that is used in this document and the *ORA Business Analytics* documents.

- **Descriptive Analytics** - uses data to understand past and current performance. It aims to answer questions such as: what happened, what is happening, how many, how often, and why? This is the most common use of traditional business intelligence tools and applications. Descriptive Analytics is most often used to support decision making, performance monitoring, and planning.
- **Predictive Analytics** - uses data, tools, and mathematical techniques to determine what is most likely to happen in the future. It applies methods such as trend analysis, pattern matching, correlation, and simulation to base future predictions

[4] Forrester Research - Overview on Business Intelligence

[5] "Competing on Analytics: The New Science of Winning", Davenport, Thomas H., and Jeanne G. Harris, 2007, Harvard Business School.

[6] Irv Lustig, Brenda Dietric, Christer Johnson, and Christopher Dziekan, “The Analytics Journey,” Analytics, Nov/Dec 2010

[7] Manan Goel, “The Art of the Possible with Business Analytics”, Analytics Powered Pursuit of Excellence

on current and historical data. Businesses use this to target products, services, and marketing campaigns, and to set forecasts for sales and earnings.

- **Prescriptive Analytics** - applies mathematical analysis to optimize business processes and operations. It prescribes the best alternative to a specific problem. Prescriptive Analytics helps answer questions such as how can we achieve the best outcome?
- **Exploratory Analytics** - is used to investigate complex and varied data, whether structured or unstructured, for information discovery. This style of analysis is particularly useful when the questions aren't well formed or the value and shape of the data isn't well understood. It can be used to discover new business opportunities, to test hypothesis regarding cause and effect, and to perform advanced forms of root cause analysis.

Given the industry support and popularity of the term "Business Analytics", the reference architecture documented in this document will use that moniker to portray both BI and analytics

Refer to *ORA Business Analytics* Documents for more details.

2.13 Information Architecture

While not currently covered in this document, this section has been included to aid in the definition and clarification of the term "Information Architecture".

2.13.1 User Experience (UX)

Information Architecture in the context of user experience (UX) focuses primarily on the structural organization/classification of digital context and interactive features to best assist users with information usablity and discovery. This includes making sure the access paths, functions, visualizations and presentations of digitized data are accessible to the disabled.

2.13.2 Enterprise Architecture

Information Architecture in the context of enterprise architecture (EA) focuses on treating Information as an enterprise asset. This includes defining principles, definitions, standards, models, and information flows to ensure that information enables information sharing and rapid business decision making.

3

Common Information Management Standards

There are many standards and patterns that can be utilized to support information management. This section describes several standards that fall under the classifications of data modeling, data representation and interoperability, data query and transformation, regulatory, and connectivity and access.

3.1 Data Modeling

There are a large number of different notations to define entities, their relationships, and flows such as Barker, Martin, UML and IDEF1X, Yourdon & Coad, and Gane & Sarson.

3.1.1 UML

The Unified Modeling Language (UML) is a standardized and extendible general-purpose modeling language which has defined 13 standard diagram types that assist with specifying, visualizing, constructing, and documenting software systems including their structure and design. UML can also be used for non-software systems activity such as business modeling.

More details can be found at http://www.uml.org/

3.1.2 IDEF1X

IDEF1X is a data modeling technique that assists in producing a graphical data model which represents the structure and semantics of information within an environment or system. IDEF1X permits the construction of semantic data models which serve to support the management of data as a resource, the integration of information systems, and the building of relational databases. The data model is unbiased toward any single application and independent of its access and physical storage.

This standard is part of the IDEF family of modeling languages in the field of software engineering.

More details can be found at http://www.idef.com/idef1X.htm

3.1.3 Crow's Feet

There are a number of notations (e.g. Barker, Martin) that utilize a popular modeling technique known as crow's feet to detail the cardinality of a relationship between two entities. Where cardinality refers to the maximum number of times an instance in one entity can be associated with instances in the related entity.

3.2 Data Representation & Interoperability

This section highlights some sample standards focusing on data representations and interoperability standards. There are a number of other commonly used data representations that are not covered here, but these representations tend not to have a fully defined standard (e.g. CSV - Comma separated values).

3.2.1 XML

Extensible Markup Language (XML) is a flexible text format derived from SGML (ISO 8879) which was designed to transport and store data and not to display data. XML describes a class of data objects called XML documents. XML documents are human-legible and reasonably clear and self-descriptive.

More details can be found at http://www.w3.org/XML/

3.2.2 XML Schema

XML Schemas express shared vocabularies and provide a means for defining the structure, content and semantics of XML documents. In effect, XML schema describes the valid format of an XML data-set. This includes what elements are (and are not) allowed at any point, what the attributes for any element may be, the number of occurrences of elements, etc.

More details can be found at http://www.w3.org/XML/Schema

3.2.3 RDF

The Resource Description Framework (RDF) is a standard model for representing information about resources in the Web while enabling data interchange on the Web. RDF provides a common framework for expressing information so it can be exchanged between applications without loss of meaning. The ability to exchange information between different applications means that the information may be made available to applications other than those for which it was originally created. It allows structured and semi-structured data to be mixed, exposed, and shared across different applications.

More details can be found at http://www.w3.org/RDF/

3.2.4 UDEF

The Universal Data Element Framework (UDEF) provides the foundation for building an enterprise-wide controlled vocabulary and is an enabler of semantic interoperability.

It is a standard way of indexing enterprise information and describing data to enable interoperability. UDEF simplifies information management through consistent classification and assignment of a global standard identifier to the data names and then relating them to similar data element concepts defined by other organizations.

More details can be found at http://www.opengroup.org/udef/

3.2.5 JPA - Java Persistence API

JPA provides support for mapping Java objects to a database schema and includes a programming API and query language for retrieving mapped entities from a database and writing back changes made to these entities. JPA enables a single API regardless of the platform, application server, or persistence provider implementation.

3.2.6 JSON

JSON (JavaScript Object Notation) is a lightweight, language-independent, human readable, hierarchical, self-describing, textual data-interchange format.

It is derived from the JavaScript scripting language for representing simple data structures and associative arrays, called objects. Data structures in JSON are based on key / value pairs. The key is a string; the value can be a numerical value, a boolean value or an object.

The JSON format is often used for serializing and transmitting structured data over a network connection. It is used primarily to transmit data between a server and web application, serving as an alternative to XML. JSON is lightweight compared to XML as no unnecessary tags are required which makes it easier to parse.

3.3 Data Query & Transformation

This section highlights a number of data query and transformation standards.

3.3.1 SQL

SQL is an acronym for Structure Query Language which is an accepted standard language for relational databases.

SQL is divided into four areas:

- SELECT statements retrieve data from the database.
- Data Manipulation Language (DML) is utilized to INSERT, UPDATE, and DELETE data.
- Data Definition Language (DDL) is utilized to create objects such as tables and indexes.
- Data Control Language (DCL) is utilized for security. DCL grants and revokes access to, and manipulation of, data by users and programs.

See the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO) for details on the latest SQL standards.

3.3.2 SPARQL

SPARQL is an acronym for Simple Protocol and RDF Query Language defines a standard query language and data access protocol for use with RDF data

See W3C for more details on SPARQL.

3.3.3 CQL

CQL is an acronym for Continuous Query Language which is a query language based on SQL with added constructs that support streaming data. Data streams represent data that is changing constantly, often exclusively through insertions of new elements.

Using CQL enables queries that support filtering, partitioning, aggregation, correlation (joins across streams), and pattern matching on streaming and relational data.

3.3.4 MDX

MDX provides a rich and powerful syntax for querying and manipulating the multidimensional data stored in OLAP server cubes.[1] It is similar to SQL in language

and structure, but includes capabilities that are beneficial for working with multidimensional data. For example, MDX includes data types for dimensions, levels (within a dimension), members (of dimensions), tuples (collection of members from different dimensions), and sets (collection of tuples).

MDX was first introduced by Microsoft in 1997. Although it is not officially an open standard, many OLAP vendors on both the server side and client side have adopted it.

MDX documentation can be found at
http://msdn.microsoft.com/en-us/library/ms145514.aspx

3.3.5 XMLA

XML for Analysis is a standard that allows client applications to talk to multi-dimensional or OLAP data sources.[2] The communication of messages back and forth is done using Web standards - HTTP, SOAP, and XML. The query language used is MDX. XMLA specifies a set of XML message interfaces over SOAP to define data access interaction between a client application and an analytical data provider. Using a standard API, XMLA provides open access to multi-dimensional data from varied data sources - any client platform to any server platform - through Web Services that are supported by multiple vendors.

XML for Analysis is designed to provide the interactive capabilities of ODBC, with support for MDX, but without the platform-specific limitations. The XML for Analysis specification is available at:
http://news.xmlforanalysis.com/docs/xmla1.1.doc

3.3.6 XQuery

XQuery is a declarative programming language that enables the ability to intelligently query and manipulate a broad spectrum of XML data stores including both databases and documents. XQuery provides the means to extract and manipulate data from XML documents or any data source that can be viewed as XML.

More details can be found at http://www.w3.org/XML/Query/

3.3.7 XLST

XSLT (Extensible Stylesheet Language Transformations) is a declarative, XML-based language used for the transformation of XML documents. XSLT is most often used to convert data between different XML schemas or to convert XML data into Web pages or PDF documents.

More details can be found at http://www.w3.org/Style/XSL/

3.4 Regulatory

This section highlights a few regulatory requirements and sample information management capabilities required to address the regulations.

[1] Carl Nolan. "Manipulate and Query OLAP Data Using ADOMD and Multidimensional Expressions". Microsoft. Retrieved 2008-03-05

[2] XML for Analysis.com

3.4.1 Sarbanes-Oxley (SOX)

Sarbanes-Oxley (SOX) is a United States federal law, created in reaction to a number of major corporate and accounting scandals. The legislation set new or enhanced standards for all U.S. public company boards, management, and public accounting firms. Sarbanes-Oxley contains 11 titles that describe specific mandates and requirements for financial reporting. SOX requires several information management capabilities such as data retention, auditing, and security.

The text of the law can be found at:
http://frwebgate.access.gpo.gov/cgibin/getdoc.cgi?dbname=107_cong_bills&docid=f:h3763enr.tst.pdf

3.4.2 Third Basel Accord (Basel III)

Basel III is a global set of measures and standards that have been defined to improve the regulation, supervision, and risk management within the banking sector. Basel III requires several information management capabilities such as data acquisition, data accessibility, governance, data quality, common reference data models, consistent views into data, and the handling of large volumes of data.

More details can be found at http://www.bis.org/bcbs/basel3.htm

3.5 Connectivity and Access

This section highlights some common connectivity and access standards used within information management technologies.

3.5.1 ODBC

Open Database Connectivity (ODBC) is a standard API method for accessing relational and non-relational data stores. It enables applications to be independent of database systems. ODBC is a client side interface used to communicate to a database on the same or different server

3.5.2 JDBC

The Java Database Connectivity (JDBC) API is the industry standard for database-independent connectivity between the Java programming language and a wide range of databases - SQL databases and other tabular data sources, such as spreadsheets or flat files. The JDBC API provides a call-level API for SQL-based database access.

Refer to http://www.oracle.com/us/technologies/java/index.html for more details

3.5.3 CMIS

Content Management Interoperability Services (CMIS) is a specification for improving interoperability between Enterprise Content Management systems. CMIS defines a domain model and set of bindings that include Web Services and ReSTful AtomPub that can be used by applications to work with one or more Content Management repositories/systems.

More details can be found at http://www.oasis-open.org/committees/cmis/

3.5.4 JSR-170 - Repository Connectivity

The Content Repository API for Java Technology is a standard repository interface for Java platforms.

The content repository API abstracts the details of data storage and retrieval such that many different applications can use the same interface, for multiple purposes. This enables a uniformed manner to access a number of content repositories.

4

Architecture Principles

This chapter contains useful architecture principles that pertain to information management. The principles are stated as rules that the reference architecture must adhere to. For each principle, a rationale is provided to explain why this principle might be desired. In addition, one or more implications are listed to provide some background on what might be involved in satisfying the principle.

These architecture principles are not intended to represent an exhaustive list; rather they offer guidance and food for thought when formulating an information management reference architecture. They have been applied to the formation of this reference architecture.

4.1 Treat All Data as an Asset

Principle	Treat All Data as an Asset
Statement	Data are an enterprise strategic asset which must be appropriately governed and managed.
Rationale	To support operational requirements and informed decision making, users, business processes, services, and systems require controlled access to the right information at the right time using the right tools.
Implications	■ All structured and un/semi-structured data must be seen belonging to the enterprise and not to an individual application or business unit. ■ Architecture must support an active and continuous upstream and downstream governance program rather than a passive downstream information governance program. ■ The enterprise must define and implement an information governance and quality program with a supporting architecture. ■ The architecture must support easy but secure access to data regardless of where it is stored or what format it is. ■ Information must be provisioned with the business user/business function in mind.

Principle	Treat All Data as an Asset
Supported By	**Conceptual View** ■ Section 5.12, "Provisioning Services" ■ Section 5.13, "Information Modeling" ■ Section 5.3.1, "Data Quality Management" ■ Section 5.5, "Analytical Data Management" ■ Section 5.9, "Data Virtualization" ■ Section 5.11, "Information Services" **Logical View** ■ Section 6.1.1, "Virtualization" ■ Section 6.3.4, "Data Warehouse"

4.2 Multiple Data Formats

Principle	Multiple Data Formats
Statement	The architecture must support a variety of structured and un/semi-structured data..
Rationale	Data held within a company are a mixture of structured and un/semi-structured. Convergence and alignment of these data formats enables better decision making.
Implications	■ The architecture must handle multiple forms of data. ■ Un/semi-structured data must be able to be tagged sufficiently in a way that supports correlation with other data forms. ■ The architecture must be able to maintain relationships between different data forms and enable navigation of unstructured data.
Supported By	**Conceptual View** ■ Section 5.4, "Historical Data Management" ■ Section 5.5, "Analytical Data Management" ■ Section 5.5.1, "Aggregation & Navagation" ■ Section 5.5.2.1, "Unification & Enrichment of Multi-Structured Data" ■ Section 5.7, "Enterprise Content Management" ■ Section 5.8, "Metadata Management" **Logical View** ■ Section 6.2.1, "Ingestion" ■ Section 6.3.4.4, "Logical Data Warehouse"

4.3 Standards Based Information Management

Principle	Standards Based Information Management
Statement	The architecture for information management must be based on a defined set of standards.

Principle	Standards Based Information Management
Rationale	Standards-based information management improves the ability to interoperate with existing but also future and unknown IT operational systems. Standards-based information management architecture supports the use of a wide range of ever changing tools for the architects and end users.
Implications	■ Support industry standards such as Web Services, JMS, XML, and JDBC/ODBC. ■ Avoid point to point integrations ■ Architecture governance process and policies should be in place to make sure agreed standards are adhered to.
Supported By	**Conceptual View** ■ Section 5.10, "Information Access" **Logical View** ■ Section 6.1.1, "Virtualization" ■ Section 6.2.1.7.2, "Hub and Spoke"

4.4 Separation of Concerns

Principle	Separation of Concerns
Statement	When there are different purposes for information, which require different structures or management techniques, then each purpose should be handled separately in order to avoid unnecessary or unacceptable compromises.
Rationale	While it is best to minimize the number of databases/repositories to manage, one shouldn't consolidate resources in such a way that compromises the important qualities and information capabilities that the business requires.
Implications	■ The architecture must support the separation of concerns between the provisioning of information and the delivery of information to consumers. ■ Authoritative, historical and analytical data should be managed and treated as separate concerns, with different requirements and priorities. This separation necessitates a mechanism to interact and provision information between these concerns such as via SOA Services. ■ Separation of historical and analytical data management does not necessitate duplication of data or data management systems, provided a single system can handle both concerns (and manage both structures) without unnecessary or unacceptable compromises. ■ Some information is managed by IT, while other information is managed by end users. The architecture must support both and provide a means to properly integrate the two.

Principle	Separation of Concerns
Supported By	**Conceptual View** ■ Figure 5–1, "High-level Conceptual Model - Separation of Concerns" ■ Section 5.4, "Historical Data Management" ■ Section 5.5, "Analytical Data Management" ■ Section 5.6, "Authoritative Data Management" ■ Section 5.12, "Provisioning Services" **Logical View** ■ Section 6.3.4.4.3, "Versatility versus Consistency" ■ Figure 6–40, "IT Managed and User Managed Data"

4.5 Volume and Velocity

Principle	Volume and Velocity
Statement	Architecture must support an ever increasing volume and velocity of data.
Rationale	Data volumes and analysis needs fluctuate along with business cycles and can grow rapidly over time. To manage these demand changes requires a flexible architecture - one that can support the scale out capacity and performance requirements to meet these demands.
Implications	■ Systems must be scalable and perform well. ■ Architecture must provide mechanisms to move large amounts of data where necessary in a timely manner due to decreasing batch windows ■ Architecture must support the separation and management of information across its lifecycle. ■ Data must be moved, transformed and loaded in an optimized manner. ■ Information management platform must evolve from batch to near real-time data ingestion by capturing changes.
Supported By	**Conceptual View** ■ Section 5.2, "Data Movement" ■ Section 5.3.2, "Information Lifecycle Management" **Logical View** ■ Section 6.2.1, "Ingestion" ■ Section 6.3.2, "Big Data Processing" ■ Section 6.3.4, "Data Warehouse"

4.6 Information Accessibility

Principle	Information Accessibility
Statement	The architecture must support various mechanisms to enable the ability to access information in a controlled and secure, but timely manner.

Principle	Information Accessibility
Rationale	Users, business processes, services, and systems require access to information to perform their functions and must not be constrained by information location, format, transport and access mechanism.
Implications	■ Support heterogeneous information consumers. ■ Support heterogeneous access protocols. ■ Support transparent integration of diverse data sources. ■ The architecture must support data federation, mediation, and consolidation. ■ Availability of information source in accordance with business needs. ■ The architecture must provide a mechanism to raise and process events to signify an update to a data source has occurred. ■ The architecture must provide a mechanism to access enterprise data via SOA Services. ■ Allowable un/semi-structured information should be able to be searched and retrieved no matter where it is in the organization.
Supported By	**Conceptual View** ■ Section 5.2, "Data Movement" ■ Section 5.2.2.1, "Change Data Capture" ■ Section 5.2.2.2, "Data Stream Capture" ■ Section 5.7, "Enterprise Content Management" ■ Section 5.9, "Data Virtualization" ■ Section 5.9.6, "Search" ■ Section 5.10, "Information Access" ■ Section 5.11, "Information Services" **Logical View** ■ Section 6.1.1, "Virtualization"

4.7 Unified View of Information

Principle	Unified View of Information
Statement	Information must be presented in a business friendly form as a single unified view rather than a collection of disparate schemas, even if multiple data sources are deployed.
Rationale	Convergence of all data into a single database is often not possible, either for technical, business, or logistical reasons. Regardless, information consumers should not be presented with duplicate and sometimes inconsistent schemas to navigate and make sense of. A single unified view offers consistency and ease of use, and it best supports the creation of a common semantic data model.

Principle	Unified View of Information
Implications	■ Present a business friendly representation of information. ■ All business data needs to have an authoritative source. ■ Analytical data must maintain conformed dimensions. ■ A virtualization layer is required if multiple data sources are deployed, and when an information request requires information spread over operational systems, historical, and analytical data stores. ■ Message and data formats should be based on logical representations of business objects rather than native application data structures. ■ Logical representations of the business entities should be based on the actual usage of the entities. ■ The architecture must provide the capability to transform data from one format to another.
Supported By	**Conceptual View** ■ Section 5.2, "Data Movement" ■ Section 5.4, "Historical Data Management" ■ Section 5.5, "Analytical Data Management" ■ Section 5.6, "Authoritative Data Management" ■ Section 5.9, "Data Virtualization" ■ Section 5.9.3, "Mediation" ■ Section 5.13, "Information Modeling" **Logical View** ■ Section 6.1.1, "Virtualization" ■ Section 6.3.3, "Enterprise Master Data Management" ■ Section 6.3.4, "Data Warehouse"

4.8 Information Abstraction

Principle	Information Abstraction
Statement	It must be possible to decouple the consumers of information from the sources of information such that changes required by consumers do not necessitate changes to data sources, and conversely, changes to data sources do not necessitate changes to consumers.
Rationale	Tools, applications, and requirements are bound to change over time. Likewise, data sources will come and go, especially when mergers, acquisitions, and IT changes occur. The architecture must be able to insulate one from the other in order to avoid a ripple effect across the system for each change that happens.

Principle	Information Abstraction
Implications	■ Support of an abstraction layer to aid in decoupling of information consumers and data sources. ■ The architecture must make no implicit assumptions about the relative locations of the information consumers and the data sources. This reduces dependency, improves flexibility and promotes agility by allowing both stakeholders to evolve and change on their terms. ■ Loose coupling of information from applications and consumers.
Supported By	**Conceptual View** ■ Section 5.9, "Data Virtualization" ■ Section 5.11, "Information Services" **Logical View** ■ Section 6.1.1, "Virtualization"

4.9 Consistent and Complete Information

Principle	Consistent and Complete Information
Statement	The system must provide a consistent and complete view of information at all times.
Rationale	It is important for the organization to understand and specify availability requirements, and for the information to be fully available during that time. This includes the period of time when data loading processes are taking place. There should be no inconsistencies or missing data due to unfinished load processes.
Implications	■ If replicating data is necessary then consumers must have access to exactly one version (release) of information at a time. A new release must be made available immediately after the previous release is taken offline. ■ The architecture may need to support queries while data loading is taking place, without the complexity of lock-escalations or the chance of reading dirty data. ■ A new version of information cannot be made available until it is fully ready (collected, cleansed, normalized, formatted, indexed, aggregated, etc.). ■ All business data needs to have an authoritative source. ■ Minimize point-to-point data integrations as they are brittle, inflexible and expensive to maintain.

Principle	Consistent and Complete Information
Supported By	**Conceptual View** ■ Section 5.2, "Data Movement" ■ Section 5.4, "Historical Data Management" ■ Section 5.6, "Authoritative Data Management" **Logical View** ■ Section 6.2.1.7.2, "Hub and Spoke" ■ Section 6.3.3, "Enterprise Master Data Management" ■ Section 6.3.4, "Data Warehouse"

4.10 Quality Data

Principle	Quality Data
Statement	Data must be trusted, fit for purpose, and be of the highest quality available.
Rationale	Operational and analytical decision making cannot be based on bad data. Therefore operational, authoritative, historical, and analytical data must be fit for purpose appropriately accurate and consistentl to reflect the facts for which they represent.
Implications	■ Data quality is seen as a continual function. ■ Data quality must be applied as near to the source as possible. ■ When possible enable data remediation by applying corrections on source systems. ■ Data quality mechanisms can be applied to both data at rest as well as data in flow. ■ Data quality indicators must be made available to consumers of information.
Supported By	**Conceptual View** ■ Section 5.3.1, "Data Quality Management" ■ Section 5.12.3, "Data Quality Services" **Logical View** ■ Section 6.2.1.5, "Data Quality Management" ■ Section 6.3.3, "Enterprise Master Data Management"

4.11 Retention of Data

Principle	Retention of Data
Statement	Facilitate the support of data retention with respect to organizations fulfilling both enterprise and industry regulatory requirements
Rationale	Organizations are subject to regulations and litigation requirements that require the retention of information for specified periods. This is driving more data to be retained, including unstructured data, and for it to be retained for longer.

Principle	Retention of Data
Implications	■ Define retention policies and dates which when applied against information, enables information to be available for the required business and/or regulatory period of time. ■ The systematic disposal of information once any associated retention period(s) expires. ■ Inhibit the systematic disposal of information for business and/or regulatory needs. (a.k.a. Freezing).
Supported By	**Conceptual View** ■ Section 5.3.2, "Information Lifecycle Management"

4.12 Flexible and Agile

Principle	Flexible and Agile
Statement	Information management must support the use of metadata to enable a flexible and agile information management platform.
Rationale	To enable a flexible and agile information management platform the architecture needs to be metadata driven. Source code modification is the least efficient mechanism to modify behavior of an information management system. The ability to change the behavior of the information management system by changing externalized metadata provides much quicker response to business needs. In addition metadata can aid in impact analysis when used in the context of insight and data lineage.
Implications	■ The metadata should be accessible both within the design-time environment (to support architects and developers) and from outside the design-time environment (to business users and data stewards). ■ Separation of configuration and other types of metadata from the source code. ■ Adopt a declarative design and model driven approach to build data assets and lay the foundation to establish an enterprise semantic data model. ■ The externalization of metadata should be incorporated into the development process. ■ Metadata driven business rules must be consistently enforced.
Supported By	**Conceptual View** ■ Section 5.8, "Metadata Management" ■ Section 5.13, "Information Modeling"

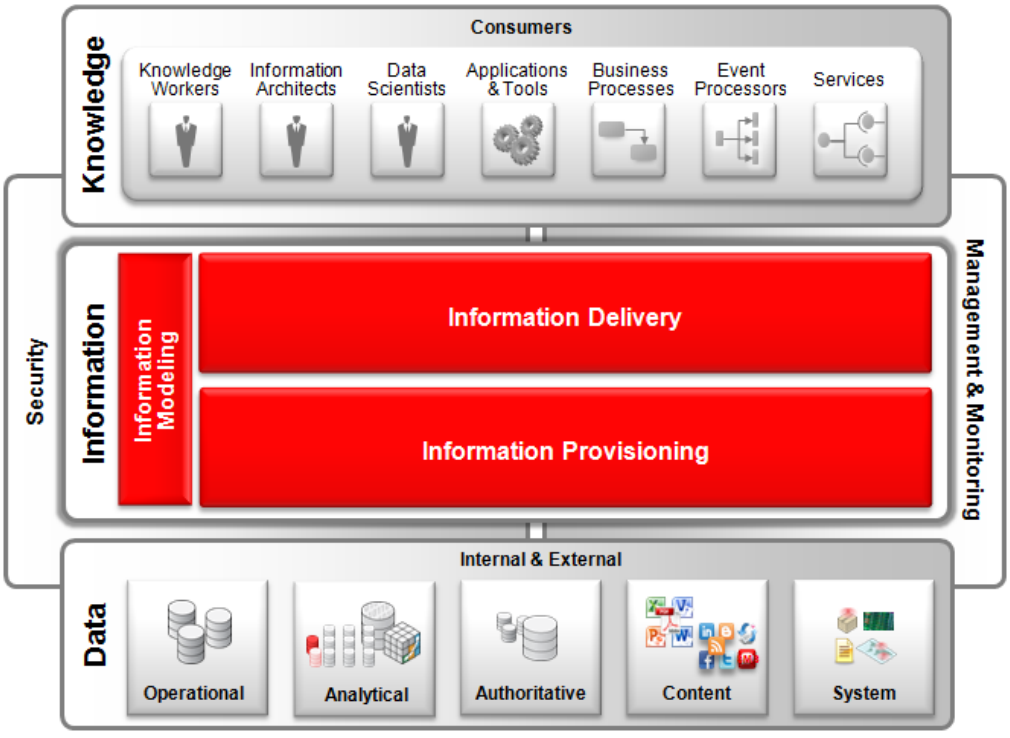
5

Conceptual View

This chapter presents a conceptual view of the ORA Information Management reference architecture. It consists of both a high level view and a more detailed view that address a set of detailed architecture capabilities.

As illustrated in Figure 5–1, "High-level Conceptual Model - Separation of Concerns", conceptually the information management reference architecture consists of three main components: Information Delivery, Information Provisioning, and Information Modeling. They draw on all the various data domains and sources of data across the enterprise, (internally and externally), and they produce information for many types of consumers (users and IT). All layers of the architecture are protected by a security architecture and are managed by a management and monitoring architecture.

Figure 5–1 High-level Conceptual Model - Separation of Concerns



Information is portrayed as an architectural layer on top of disparate data domains. It organizes and presents all types of raw data in a way that maximizes business value, integrity, and usefulness. Data in this diagram refers to various data domains and sources of data that require to be provisioned for universal use in a consistent and

trusted manner specifically for consumers such as executives, planners, applications, tools, business processes, and services.

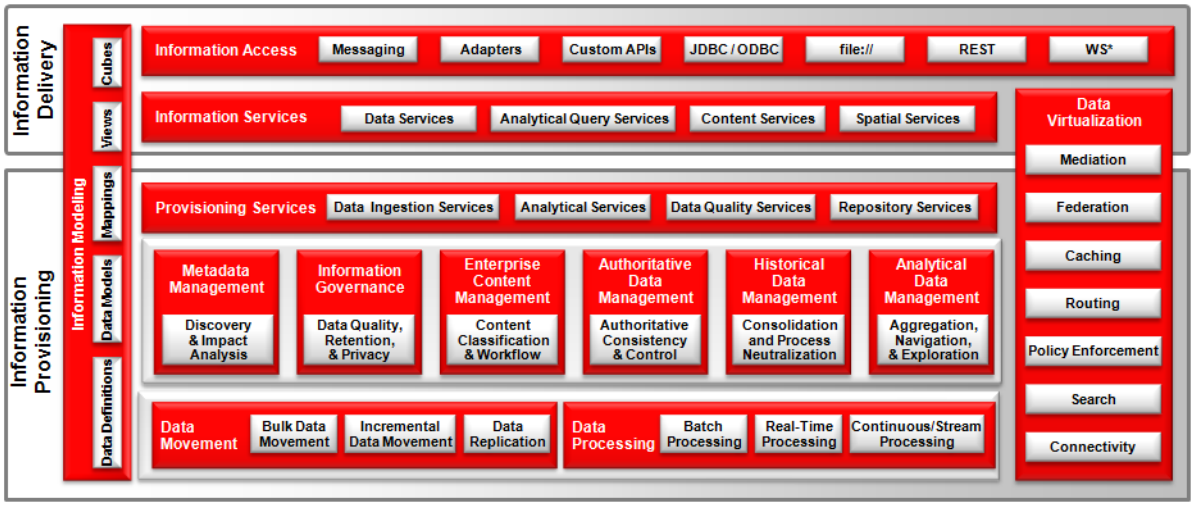
The Information Provisioning Layer provisions information in various forms and semantics using industry and enterprise models and for various consumers, e.g. it is not tied to a specific solution or consumer; rather it is designed for broader use and multiple purposes. The Information Delivery Layer organizes, and delivers information in a way that best represents the needs of the consumers, (e.g. semantics and access).

This separation of concerns between Information Provisioning and Information Delivery insulates consumers from the complexity and location (internal and external) of the underlying data. In addition, this level of abstraction allows enterprises the flexibility to change the underlying data sources and schemas without impacting consumers.

All too often, individual projects implement point solutions to address their specific project's requirements without considering the wider implications of both security and management & monitoring. This should be addressed by a strategic approach that should be based on a solid, holistic framework encompassing all of an organization's Information Security and Management & Monitoring requirements, not just those of individual projects. Please consult *ORA Security and ORA Management & Monitoring* for more detail in these areas.

A more detailed view of the architecture is provided in Figure 5–2, "High-level Capabilities". Here the Information Delivery and Information Provision Layers have been expanded to drill down one more level of detail illustrating the high level capabilities.

Figure 5–2 High-level Capabilities



The following subsections describe each layer of the architecture in detail in terms of the capabilities it provides.

5.1 Data Processing

This section focuses on a number of different architectural techniques and capabilities in the processing of data. Data processing can occur in batch, real-time, and stream modes.

There are a number of data processing capabilities that address specific challenges (e.g. incremental computation on large data sets, graph processing utilizing bulk synchronous parallel), but they are not currently covered in this document due to their lack of maturity and adoption.

Figure 5–3 Data Processing Capabilities

5.1.1 Batch Processing

Batch processing involves the processing of data at specific intervals, asynchronous to when results are needed. Batch jobs are often initiated by a scheduler or via human interaction. The results are maintained and made available to users, processes, or applications that need them. As such, they can be considered "long-lived", e.g. the results will not change until the next batch processing run occurs.

Batch processing is often used when working with very large data sets or in near real-time when results need not reflect current, frequently changing conditions. As discussed earlier in the document, organizations have access to vast amounts of data from both current and new data sources (e.g. content and system data). In the past it has not been practical to integrate this data, (especially machine-generated data), into an overall information management strategy. Due to the infrastructure costs, the challenges arising from the scale of the data (e.g. volume, velocity), the varied formats (unstructured and semi-structured), and the complex techniques needed to process it.

To address these challenges, a number of parallel batch processing models have been defined. The current popular choice for distributed parallel data processing is a programming model called MapReduce. MapReduce is a shared-nothing data processing architecture, and parallel programming model, that supports processing of large data sets on distributed nodes while hiding the complexity of distribution and fault tolerance. In effect, MapReduce divides a large processing task into smaller tasks, processes them in parallel, and then combines the output.

MapReduce supports a wide range of data processing capabilities for large data sets such as ETL, transformation, data cleansing, consolidation, and aggregation. This assists in addressing requirements such as generating inverted indexes for text searching, and the analysis of non-relational data such as log files. MapReduce is not suitable for all problems. New architectures and programming models are always being created to address specific data processing requirements such as incremental data processing and graph processing.

5.1.1.1 In-Database Processing

There are many approaches to performing batch data processing within a database. The most prominent are procedural SQL and Java. The ability to perform data processing within the database enables the utilization of its inherent and efficient parallel capabilities.

One area of increasing interest is the ability to run MapReduce applications inside a relational database (a.k.a. In-Database MapReduce (IDMR)). This approach efficiently integrates the MapReduce programming model with a parallel relational database. A separate MapReduce environment is not always needed as the amount and structure of the data enterprises have are not always on the scale as large social web sites

In addition, this coalescing of MapReduce and relational database capabilities brings additional advantages:

- It provides a versatile solution which allows organizations to use their existing programming and administration skills.
- It provides access to data without the need to move the data to separate, potentially insecure MapReduce infrastructure.

5.1.2 Real-time Processing

Real-time processing, in this context, refers to processing that occurs "on demand". Each request will be processed when it occurs and the results will be returned once processing completes. Real-time processing supports in-line analytical operations, such as recommendation engines, real-time pricing algorithms, and decision logic. Results are typically "short-lived" as there is generally a direct correlation between when results are needed and when processing occurs. But results may be cached and/or saved for future use and analysis.

5.1.2.1 In-Memory Processing

In-Memory processing enables data processing to occur where the data is located within the data grid. This leads to extremely fast and scalable data processing capabilities. The data grid is topology-aware and determines to which node(s) the processing agent needs to be located. Each processing agent then executes against the data held locally in memory.

This is a very efficient approach as no data other than the processing agent itself needs to be moved across the network, and each node only needs to process a small subset of the data. By executing processing agents in parallel against all nodes in a grid, large amounts of data can be processed and aggregated in a MapReduce manner.

5.1.3 Continuous / Stream Processing

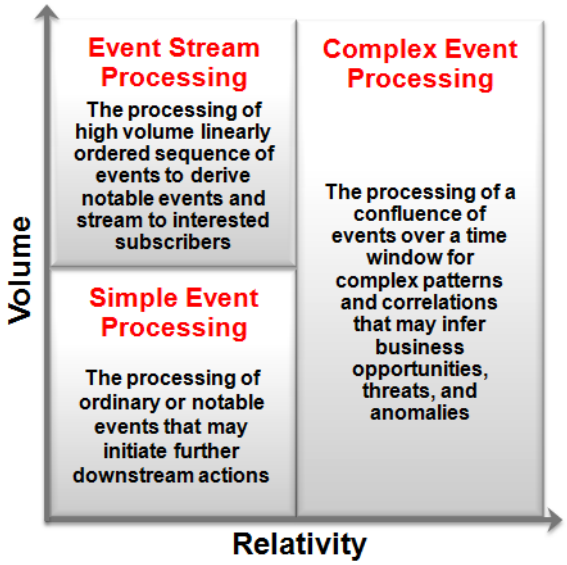
Continuous / Stream processing is used when data needs to be processed constantly. A continuously running process operates on data (aka event data) as it is received. The process may be designed to take specific actions depending on data content or context.

The stream of data that is processed can be considered as a stream of events as the data has been generated to an event occurring. Therefore this form of stream processing is also referred to as event processing.

Individual low-level events are rarely actionable, in that they tend not to be significant enough for the business to react and take a meaningful action upon. This is especially true for streaming and high volume events. These events need to be further processed to derive actionable events. Event processing manipulates events through the application of business rules to identify opportunities, threats, and anomalies that need to be acted upon.

Event Processing filters and manipulates events to derive useful and actionable business events. Rules/queries are used for processing an inbound stream of events, eventually generating an outbound stream of events. Generally, the number of outbound events is much lower than that of the inbound events due to elimination and aggregation. There are various classifications of event processing based upon volume and the relativity between the individual events (See Figure 5–4, "Event Processing Classifications").

Figure 5–4 Event Processing Classifications



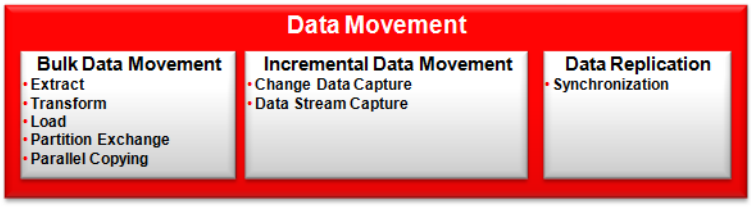
Event Processing is rarely a one-step process. Many times a multi-step approach is required where events can be filtered, enriched, correlated, aggregated, and patterns can be recognized. Sometimes it is necessary to use contextual or historical information from internal sources such as databases (relational or NoSQL) to derive and enrich the outbound events. For example, in the financial industry, the credit card processing events may be combined with the stolen or suspect credit cards information from the internal sources to generate fraud detection events.

Refer to *ITSO EDA documents* for more information regarding event processing.

5.2 Data Movement

Data Movement focuses on a number of different architectural techniques and capabilities in the movement of data from one provisioned data source to another. This category of capabilities covers batch and real-time data movement as well as bi-directional replication and synchronization.

Figure 5–5 Data Movement Capabilities



5.2.1 Bulk Data Movement

Bulk Data Movement of structured data has traditionally been associated with a process known as ETL (Extract, Transform and Load) but more recently a more effective and scalable approach has been utilized known as E-LT (Extract, Load, and Transform).

5.2.1.1 ETL (Extract, Transform, Load)

ETL is a middle-tier engine centric architecture and is the more traditional approach to data movement. ETL initially **extracts** the data from various source data stores, then **transforms** the data in a middle-tier ETL engine within a staging area, and then **loads** the transformed data into the target data store.

The data transformation step of the ETL process is by far the most compute-intensive, and is performed entirely by the ETL engine. The ETL engine performs data transformations (and sometimes data quality checks) on a row-by-row basis, and hence, can easily become the bottleneck in the overall process.

In addition, the data must be moved over the network twice - once between the sources and the ETL server, and again between the ETL server and the target data store.

5.2.1.2 E-LT (Extract, Load, and Transform)

E-LT is database centric architecture that provides a flexible and optimized approach to data movement. There are several advantages to this E-LT approach such as utilizing parallel processing features of the database to perform the transformation and reuse of common in-house skills such as SQL. In addition if the staging area and target data source are located on the same database then less data needs to cross the network.

5.2.1.3 Logical Data Movement

Apart from physical Data Movement capabilities such as EL-T and ETL some consideration should be given to logical data movement capabilities such as partition exchange.

Partitioning allows a table, index, or index-organized table to be subdivided into smaller pieces, where each piece of such a database object is called a partition. By manipulating partitions in your target data store, you can use Partition Exchange to instantly move data. Partition Exchange has the effect of moving data simply by swapping definitions between two data partitions. It appears to have moved even though no transmission of data has occurred.

5.2.1.4 Parallel Copying

Moving a large amount of files in a sequential manner can take a considerable amount of time. Parallel copying is the capability to parallelize copying of large amounts of data therefore speeding up the process. This capability is applicable for both native and distributed file systems.

5.2.2 Incremental Data Movement

Informed decision making requires access to more real-time data. As the latency of data movement increases it starts to lose relevance to ongoing operations, and thus its value decreases.

5.2.2.1 Change Data Capture

Change data capture (CDC) plays a key role in keeping data consistently updated without overly impacting the target or source performance.

Change Data Capture is accomplished by capturing just the changed records and not the full data set and delivering this to the target data store, thus dramatically reducing time and resources for the movement of data. While there are different methods to capture only change data from source systems (e.g. log based, table trigger based),

reading transaction logs (log based CDC) has the lowest impact on the source systems, in addition to being non-intrusive.

For maximum performance (speed and scalability), it is important that CDC provides low impact capture, routing, transformation, and delivery of transactional data across heterogeneous environments in real time without the need for an ETL engine.

Any CDC approach must guarantee referential integrity. Changes frequently involve several data stores at one time. For example, when an order is created, updated, or deleted, it involves both the orders table and the order lines table. When processing a new order line, the new order to which this line is related must be taken into account.

5.2.2.2 Data Stream Capture

Data Stream Capture (DSC) addresses the need to lower latency when the latency of batch movement of unstructured/semi-structured data is unacceptable.

Data Stream Capture provides the capability to move data as it is generated by describing the way a single stream of data is delivered and processed from its source to its target data store (e.g. distributed file system). Streaming data are captured via a number of mechanisms (e.g. tail of a text file) and then transmitted to their destination.

Multiple Data Stream Capture engines can be wired together to support a wide variety of data movement requirements (e.g. scalability, routing, consolidation, and reliability).

Data Stream Capture not only addresses low latency requirements but also provides a low impact capture, routing, transformation, and delivery of un/semi-structured data across heterogeneous environments in real time.

5.2.3 Data Replication

Data Replication enables the creation and maintenance of one or more synchronized standby data stores that protect data from failures, disasters, errors, and corruptions. It can address both High Availability (HA) and Disaster Recovery (DR) requirements. In addition, it can address off-loading reporting requirements from the production data stores.

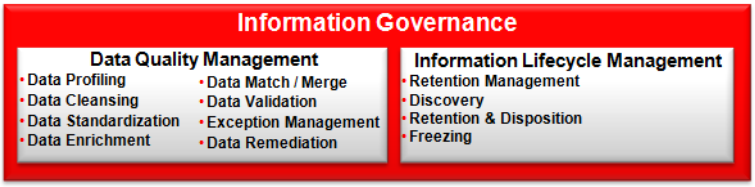
When the data being sourced and targeted are the same, and requirements are focused on high availability and disaster recovery, then a Data replication approach is best suited. If the source and target sources are heterogeneous then a combined bulk data movement and incremental data movement approach is best suited.

A standby data store is initially created from a backup copy of the primary data store and then the replication engine keeps the data synchronized.

5.3 Information Governance

This set of capabilities supports an overall information governance strategy, which includes data quality management and information lifecycle management. As highlighted in the preface, this document focuses on the architectural aspects of Information Governance, not on the process and organizational aspects.

Figure 5–6 Information Governance Capabilities



5.3.1 Data Quality Management

As previously highlighted a critical function of information governance is to ensure data quality. Data Quality Management is an approach in making and keeping data in a state of completeness, validity, consistency, timeliness, and accuracy, that makes it appropriate for a specific use.

5.3.1.1 Data Profiling & Validation

To lay the foundation for an overall data profiling approach, this capability assesses and identifies common data defects. Along with profiling metrics, rules and actions for fixing data are defined. This enables both an automated and manual approach to data profiling.

Data Profiling & Validation provides an initial baseline for understanding the ways in which actual data values in data stores fail to conform to expectations with regard to quality, usefulness, and conformance to rules and standards.

Initially metadata is discovered and reverse-engineered from various data stores. Then data profiling analyzes existing data by identifying, categorizing, and qualifying low-quality data. A number of profiling characteristics are utilized to analyze areas such as attribute lengths, minimum and maximum values, unique values, value distributions, data types, and data relationships. See Table 5–1.

Table 5–1 Sample Data Profiling Metrics

Characteristic	Sample Profiling Metric
Completeness	▪ Null Counts ▪ Minimum & Maximum Lengths
Conformity	▪ Field Structure ▪ Data Types ▪ Patterns / Masks
Validity	▪ Unique Values ▪ Specific Business Rules
Consistency	▪ Soundex ▪ Metaphones
Integrity	▪ Dependencies ▪ Keys ▪ Joins

Examples profiling characteristics include:

- **Completeness** - utilized to assess that all necessary data are present checking for areas such as null attributes and minimum and maximum lengths.

- **Conformity** - utilized to check the level of conformity and standardization of the data in accordance to defined patterns/mask rules and reference dictionaries as deemed by the business/industry.
- **Validity** - utilized to identify invalid data that are not fit for purpose. Examples include validating social security numbers, vehicle identification number, product data, brand data, and financial data. A common example is that of an invalid address. Addresses can be validated against national postal authority files. If no reference data are available then a frequency distribution approach is often used whereby a count is made of the distinct values in a column. Often, if there is a value with only one occurrence in a table with millions of rows, it may be an indication that this value should be merged with other values.

  As well as checking data for validity against reference sources data, attributes can also be validated to make sure that they are within a boundary of approved values. For example, a birth date may be checked to verify if the person is no more than 150 years old, or a percentage measurement must be in the range 0 to 100.
- **Consistency** - utilized to assess the consistency of data across many systems. For instance, techniques such as soundex and metaphone to identify similar sounding words even though they are spelled differently. This assists with matching despite the words having minor differences. Another example is checking the consistency of calculations across multiple systems.
- **Integrity** - utilized to assess the integrity of the data by analyzing keys, joins and dependencies. This can be further complicated by poor schema design or sourcing data from a data source that does not have integrity checking built in, i.e. text files.

Data policies define the rules that assist with profiling and assessing the quality of the data. Here are some example rules:

- Uniqueness Rules
  - Different customers must not have the same e-mail address
  - Different products must have different product and family codes
- Simple and complex reference rules
  - All customers must have a sales representative
  - Orders must not be linked to customers marked as 'Invalid'
- Validation rules that enforce consistency at the record level
  - Customers must not have an empty zip code
  - Web contacts must have a valid e-mail address

As part of a continuous exercise, data profiling enables the relevant business/IT user to review and confirm a data quality improvement over time.

5.3.1.2 Data Cleansing

Data Cleansing also known as data scrubbing is the act of correcting data errors. Data Cleansing is often performed as a set of automated steps that can correct common errors, along with manual intervention as needed. For example data entry errors invariably will lead to a number of text columns contain mis-spelt text e.g., femail. A level of data transformation will be required to support any changes in structure and representation of the data.

Once common data errors have been addressed the data cleansed can be later standardized if necessary.

5.3.1.3 Data Standardization

Invariably, data are stored in source systems in different formats, therefore a standardization of the data are required when accessing, moving and/or consolidating the data. Data Standardization adjusts data formats and values so that records of diverse representations are all represented using common and/or enterprise standards. Examples include standardizing the format of a telephone number, an address, or a unit of measure,

Standardization efforts can be based on differing structures that conform to defined rules, such as:

- Text structure (First Name + Last Name vs. Last Name + First Name)
- Date Structures (dd/mm/yyyy vs. mm/dd/yyyy)
- Value Structures (Male, Female vs. M,F)

These rules can be flexible. For instance, contact and address information can be standardized based on the country of origin, or according to corporate standards.

5.3.1.4 Data Enrichment

Data Enrichment adds additional values and metadata tags to existing data in a controlled manner as to override manual augmentation. This capability can address data quality issues such as replacing missing values by either inserting defaults or using 3rd party syndicated data. For example, a syndicated global name and address database can be utilized to fully complete the address fields. In addition, geographic or third-party information (latitude and longitude, or market targeting information) could also be added to the address information.

5.3.1.5 Data Match/Merge

Data Match/Merge detects, links, and merges similar data, in effect performing a data de-duplication whereby a "best fit" record is defined and all duplicates are linked to it. This capability may involve aspects of validation, profiling, cleansing, standardization, and enrichment in order to deduce a possible match. For example, two products in different languages are in fact the same item. 'Like' records that are matched and merged are reconciled through survivorship rules. This provides the ability to define that data from certain sources are more trusted than others.

5.3.1.6 Exception Management

Not all data quality issues can be handled in an automated manner using policy rules and data quality engines. Therefore a data quality capability must support the handling of all data issues that cannot be automatically processed for quality. Data that have issues are often staged for manual processing as part of oversight and exception management.

Decisions will need to be made regarding what to do with data that has failed validation. They can be ignored, quarantined, or be allowed and labeled as dirty data.

Data that does not match authoritative values may be 'orphaned' until the parent values are added or corrected at which time an automatic 'orphan adoption' process can occur after which the data is no longer labeled as failing validation.

5.3.1.7 Data Remediation

Also known as Data Harmonization, Data Remediation is the ability to push the now clean data back into the source data stores. This has the following advantages:

- Source systems have access to the improved data.

- Cleaning work does not need to be redone during future data movement processes.

While it is best to remediate cleansed data into the source system, it is not always possible. Restrictions in the source system may prevent the cleansed data from being remediated.

5.3.2 Information Lifecycle Management

As stated in the Concepts chapter, Information Lifecycle Management (ILM) comprises of the policies, processes, practices, and tools used to align the business value of information with the most appropriate and cost effective infrastructure from the time information is created through its final disposition.

Companies taking an Information Lifecycle Management approach recognize that the importance of any information does not rely solely on its age or how often it's accessed. Therefore, a systematic policy-based approach is needed whereby information is governed and managed throughout each phase of the information lifecycle. Therefore, organizations need to understand how their information evolves, determine how it grows, monitor how its usage changes over time, and decide how long it should be retained, while adhering to all the rules and regulations that apply.

5.3.2.1 Retention Management

Organizations have long achieved archiving and retention management around structured data using capabilities built into enterprise-grade relational databases. But content and system data are growing exponentially across numerous repositories, such as content management, file systems, and email servers, which are adding complexity and higher storage costs. At the same time, organizations are subject to ever increasing regulations and litigation that require the retention of this information for a specified period of time.

Retention Management assists with controlling these costs and risks, as data that is no longer useful or required is systematically destroyed. This can minimize litigation risks and discovery costs by reducing the amount of data that must be audited and reviewed.

5.3.2.1.1 Discovery Discovery provides the ability to search and retrieve all of the important content within an organization no matter where it is in the organization. Discovery makes it easier to locate information during legal or audit discovery. Discovery can also be executed in a federated fashion whereby content across all repositories can be searched and retrieved. This allows organizations to leave content in its existing location, rather than moving it to a central repository for retention management.

5.3.2.1.2 Retention & Disposition Managed items are assigned retention schedules and disposition rules which enable:

- Outdated or superseded information to be eliminated.
- The management of storage resources.
- Compliance with legal and/or regulatory requirements.

Information may need to be retained for different periods of time, depending on the type of content, its use within the organization, and the need to comply with external laws or regulations. The retention period may be:

- Time-based (for example, five years from the filing date).

- Event-based (for example, an employee termination).
- Both time-based and event-based (for example, two years after employee termination).
- Based on usage, if usage is being tracked.
- Based on revision (for example, after four revisions).

After a retention period, eligible content is systematically disposed of. Some content is deemed so important it will never be destroyed (for example, due to historical significance)

5.3.2.1.3 Freezing Freezing inhibits disposition processing for an item. Frozen content cannot be altered in any way, nor can it be deleted or destroyed. This may be necessary to comply with legal or audit requirements (for example, because of litigation).

Federated Freezing inhibits disposition processing for items that are located in other parts of the organization. This enables organizations to leave content in its existing location, rather than moving it to a central repository for records and retention management.

5.4 Historical Data Management

Historical Data Management is designed to accurately maintain a history of data. Structured data are gathered, consolidated, normalized, and versioned in a process-neutral form. The ability for end users to navigate data in this form is of less concern. Versatility, consistency, depth, and accuracy, are paramount.

For less structured data, historical data management also maintains a history of the data which has been preprocessed, but due to its structure, has not been normalized. This data can be used for later exploration or analysis.

Figure 5–7 Historical Data Management Capabilities

The following Historical Data Management capabilities are primarily focused on structured data. Aspects of these capabilities could be applied to less structured data but tend to be addressed by other information management capabilities. For example, Versioning of documents can be addressed via capabilities within Enterprise Content Management.

5.4.1 Atomic / Process Neutral

Organizations and business processes change over time, therefore for greater flexibility and adaptation, historical data are maintained in an application- and business-neutral form. This allows the structure to better support changes to business processes and hierarchies without the need for schema changes.

5.4.2 Data Consolidation

With data typically fragmented and duplicated across operational silos, it is often difficult to determine in which version, and in which system, the data entity is most accurate and complete. Data consolidation resolves these issues by integrating and consolidating data from across the enterprise. This allows the deployment of a single consolidation point that spans data formats, integration modes, technologies, and standards, which enables the enforcement of consistency and standardization.

Data consolidation acquires, maps, and cross-references data from multiple source systems. At the same time it invokes data quality management capabilities to standardize, cleanse, eliminate duplication, and produce a blended record, taking into account survivorship rules and source system priority rankings for each attribute.

5.4.3 Data Normalization

Relational Database Management Systems (RDBMS) are designed to store structured data in tables that support relationships between tables in the form of keys. In order to maximize efficiency, and minimize redundancy, data are usually "normalized". There are different degrees to which normalization can be applied. The benefits of various degrees of normalization must be weighed against the added complexity they bring. Sometimes data are intentionally "de-normalized" in order to improve system performance and make the schema easier to understand.

The most common degree of normalization is known as 3rd normal form (3NF), which discards repeating groups, minimizes redundancy, eliminates composite keys for partial dependency, and separates non-key attributes. 3NF not only reduces the total amount of data that needs to be stored, but also promotes consistency and makes it easier to maintain. A change to customer information, for example, would involve changing only one entry in the Customer table vs. many entries in a combined Customer/Sales table.

5.4.4 Data Versioning

There are several well-recognized approaches for managing changes to data for historical reporting purposes[1]. The following example scenario is used to highlight these various approaches.

Mary Smith marries Bob Jones and decides to change her name to Mary Jones:

- **Type 1: Replace the value.** In the case of a last name change, the old value is simply overwritten with the new value. This method makes sense when changes have no effect on the value of historical consistency.

Figure 5–8 Data Versioning Type 1 - Before

Surrogate Key	First Name	Last Name
12345	Mary	Smith

In our example the fact that Mary was once Mary Smith is lost. See Figure 5–9, "Data Versioning Type 1 - After" below.

[1] Slowly Changing Dimensions - Kimball Group

Figure 5–9 Data Versioning Type 1 - After

Surrogate Key	First Name	Last Name
12345	Mary	Jones

- **Type 2: Add a record with an effective start date and effective end date.** This solution preserves history by maintaining both values in separate records. The date fields are used to determine which value is in effect at a particular time.

  In our example, a new record for Mary Jones with an effective start date is created. The original record has the effective end date updated with the same date. The fact that Mary was once Mary Smith is not lost.

Figure 5–10 Data Versioning Type 2

Surrogate Key	First Name	Last Name	Effective Start Date	Effective End Date
12345	Mary	Smith	1/18/1960	3/14/2006
45678	Mary	Jones	3/14/2006	

- **Type 3: Store the old value.** This solution provides some backward reference by maintaining historic values in "previous value" fields. Only one record is maintained, and the number of historic values is often limited to one. It is most useful when historic values provide informational value, but are not necessary for computational value.

  In our example the prior last name field has been updated to Smith, while last name field has been updated to Jones.

Figure 5–11 Data Versioning Type 3

Surrogate Key	First Name	Last Name	Prior First Name	Prior Last Name
12345	Mary	Jones	Mary	Smith

- **Type 6: A combination of Types 1, 2, and 3.** In this case the current value is replaced, a previous value is maintained, and a new record is added with start and effective dates. This option provides the most information, but has the highest cost in terms of storage and maintenance.

  Expanding on the example, let's say that Mary Smith was born on January 18th 1960. See Figure 5–12 below.

Figure 5–12 Data Versioning Type 6 - Before

Surrogate Key	Current First Name	Current Last Name	Historical First Name	Historical Last Name	Effective Start Date	Effective End Date
12345	Mary	Smith	Mary	Smith	1/18/1960	

Mary Smith marries Bob Jones and becomes Mary Jones on March 14th 2003. See Figure 5–13 below.

Figure 5–13 Data Versioning Type 6 - Change 1

Surrogate Key	Current First Name	Current Last Name	Historical First Name	Historical Last Name	Effective Start Date	Effective End Date
12345	Mary	Jones	Mary	Smith	1/18/1960	3/14/2003
45678	Mary	Jones	Mary	Smith	3/14/2003	

On July 24th 2005, Mary Jones divorces Bob Jones. On September 17th 2005, Mary Jones marries Mark Davies and chooses to take his last name. See Figure 5–14 below.

Figure 5–14 Data Versioning Type 6 - Change 2

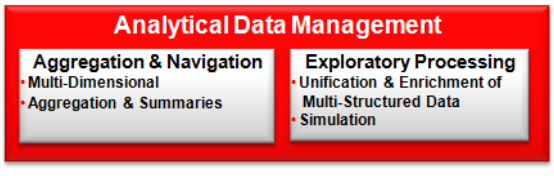
Surrogate Key	Current First Name	Current Last Name	Historical First Name	Historical Last Name	Effective Start Date	Effective End Date
12345	Mary	Davis	Mary	Smith	1/18/1960	3/14/2003
45678	Mary	Davis	Mary	Jones	3/14/2003	7/24/2005
56789	Mary	Davis	Mary	Jones	9/17/2005	

5.5 Analytical Data Management

Even though Historical Data Management has its role and advantages as a storage mechanism, it is not very helpful when giving access to the information to business/data analysts due to the fact that data are either in 3rd normal form or pre-processed unstructured form. It lacks the complex aspects of business rule interpretation and makes it more difficult to navigate. Information must be available in a form that allows analysts to understand it and navigate it easily.

Analytical Data Management enables current and historical information to be easily navigated and efficiently queried and summarized. For navigation of structured data, a multi-dimensional representation is utilized to reflect the current state of business process, hierarchies, and entities. It may also include prior states to enable comparative reporting. For less structured data, analytical data management provides a persistence area for snapshots, samplings, or subsets of data that can be used for exploration or analysis that does not need to be executed across the entire history of data.

Figure 5–15 Analytical Data Management Capabilities



Analytical data may also include user-defined data such as forecasts, segments, attributes, and simulation variables that may factor into analytical reports, quotas,

projections, etc. It may be managed centrally, by an IT department, or locally, by end users.

Refer to the *ORA Business Analytics* document for more details.

5.5.1 Aggregation & Navagation

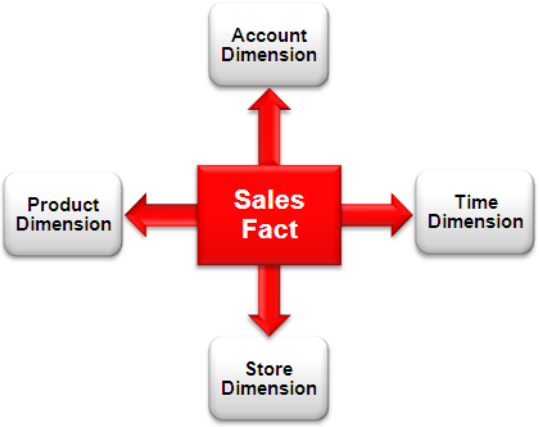
5.5.1.1 Multi-Dimensional Representation

Business users understand how the business is organized, not how a database is organized. Complex relationships, joins, and nesting make sense to a data architect but not so much to anyone else. In order to sift through data effectively the business user needs to view data more simply and hierarchically, in a form that best mimics the way the business operates.

As previously highlighted, dimensional modeling is a method of organizing data in a way that represents natural business hierarchies.

Business events can be linked to several different dimensions, as shown in Figure 5–16. For example, an item being sold can be linked to dimensions that represent customer account, product, location, and time. The event or measurement of what happened is called a fact. Facts are generally stored in tables that link to the dimension tables via a collection of logical foreign keys.

Figure 5–16 Sample Dimensional Model



There are several forms of dimensional modeling that are variations or extensions on this theme. They include star models, snowflake models, galaxy models and constellation models. Refer to the *ORA Business Analytics* Documents for more details on Dimensional Modeling.

Utilizing OLAP, multi-dimensional queries define the subset of dimensional data to be analyzed. They define the dimensions and levels from which to results sets are returned. For example, a query might ask for all sales data from store 123 pertaining to brand X for the previous six months. The query might also ask to compare these results with the same data from the previous year.

Refer to the *ORA Business Analytics* Documents for more details on OLAP.

5.5.1.2 Aggregation & Summaries

Given the frequent need to summarize data for meaningful comparisons, it makes sense to consider pre-calculating some of the totals. This can drastically improve query performance and reduce the processing power needed to handle OLAP functions. The summarized values are called aggregations.

Depending on the anticipated reporting needs, aggregations can be created and stored for any or all of the dimensional hierarchy levels. For example: sales by store, district, or region; sales by product, brand, or category; and sales by day, month, quarter, or year. Aggregations can also combine dimensions, e.g. sales by product, quarter, and region.

In terms of architecture, the system must be able to determine when aggregates are available to help satisfy a query, even when they are not the full answer to a query. For instance, if a user requests sale figures for the year, and aggregates are available at the month level, then month aggregates should be summarized to produce a result. The system must recognize the most efficient means to produce a result and automatically alter the query path accordingly.

Aggregations are very powerful, but there is a trade-off between run-time performance and "load-time" preparation and storage. Whenever any new data are added to the model or when hierarchical relationships are changed, all (affected) pre-summarized data values must be recalculated. In addition, summary data must also be stored, thus increasing overall storage requirements.

It is important to consider several factors when developing an aggregation strategy, such as:

- The size of fact tables and distribution of facts across dimensions. Aggregation may be necessary when a large number of facts are concentrated on specific dimension elements.
- The duration of a query without summarization.
- The frequency a given summary is typically used.
- The frequency that data are added to the model.
- Whether the aggregates are to be stored close to the atomic data.
- The time available to perform aggregations. This takes into account factors such as data loading time, frequency, and the uptime requirements of the system (e.g. 7x24 vs. 5x9).

An alternative to pre-aggregation is post-aggregation. With this method summaries are not created in advance, rather they are calculated when the first query is performed, and subsequently stored for use on future queries.

5.5.2 Exploratory Processing

Exploratory Processing applies advanced ad-hoc exploratory activities that generally involve the analysis, filtering, distillation, classification, and correlation of data in order to create a useful form of knowledge or understanding. This assists users in attaining more from business analytics investments by enabling intuitive exploration and analysis of information from complex and varied data (e.g. any combination of structured, semi-structured, and unstructured sources).

Exploratory Processing lays the information provisioning foundation on which Exploratory Analysis tools can provide visibility into any source and more importantly, allows analysis alongside traditional business intelligence source systems.

This results in time and cost savings and enables improved business decisions based on a deeper understanding of the business.

5.5.2.1 Unification & Enrichment of Multi-Structured Data

Analysis may involve a combination of structured, semi-structured, and unstructured data. It can be extremely costly and time consuming to making these radically different and constantly changing data types fit within a rigid data model. Therefore, a defined or rigid data model is not appropriate for this form of analysis.

Additionally, the very nature of performing exploratory analysis is that users do not know what questions to ask next. This makes it difficult to pre-define a data model for exploratory analysis.

What is required is a dynamic, faceted data model that evolves depending on the data being ingested. A faceted data model is extremely flexible and reduces the need for up-front data modeling. It supports an iterative "model as you go" approach.

As this varied data is ingested it can be stored, enriched and organized. Data is unified via derived common attributes from attached metadata. For example, structured data may be ingested from a Data Warehouse. The attributes are derived from the column names of the table. If unstructured data is also required to be ingested then advanced text enrichment techniques can be applied to extract structured data, such as people, places, subjects, and sentiment, from the raw text and tagged on to the existing faceted data. The data now contains a combination of structured data and textual attribution, enabling business users to explore and analyze the data in ways never before possible.

5.5.2.2 Simulation

Historical data can be utilized to forecast future performance or events. Organizations can attempt to create forecasts based on projecting current trends, by modeling simulations where various data points are purposefully altered, or by detecting hidden patterns that affect operations.

When simulation requires adding and altering data values, it should be performed on a snapshot of the historical data, where changes can be easily made and alteration will not compromise the integrity of the underlying data.

5.6 Authoritative Data Management

It has become increasingly difficult for organizations to identify, maintain, and use an authoritative set of master and reference data in a consistent way across the enterprise.

In an ideal world, there would be a single place where all common master and reference data in an organization is stored and managed. The data would be accurate, consistent, and maintained in a coherent and secure manner. All updates would take place against this single copy of data, and all of the different users of master data would interact with this single authoritative source of information. This would require that all applications would utilize this single source. In reality, enterprises have multiple and inconsistent definitions of authoritative data for various reasons, e.g. lack of a comprehensive data governance repository detailing data currently held by the enterprise, different channels, packaged applications, mergers and acquisitions, siloed IT department funding.

Figure 5–17 Authoritative Data Management Capabilities

Authoritative Data Management enables the central source for accurate, fully cross-referenced, real-time, authoritative data. It consolidates master and reference data from all connected applications and standardizes on a central data model. In addition, it creates an authoritative "golden" record that represents a single version of the truth about each and every authoritative data entity.

5.6.1 Multi-Entity Management

While there is value in organizations addressing one authoritative entity, over time organizations will face the need to address an ever increasing number of authoritative master and reference data entities to support their applications, business processes, and analytical needs.

An information management architecture must be able to support a single version of the truth for every authoritative data entity required by the enterprise. This requires not only being consistent at the entity level, but also in the relationships between entities.

5.6.2 Common Authoritative Data Model

Authoritative Data Management makes use of a common authoritative data model for both the storage and transformation of authoritative data.

The model supports all the authoritative data that drives a business, no matter what systems source the authoritative data fragments. The authoritative data model is unique in that it represents a superset of all the ways authoritative data has been defined by all attached operational systems. The model is defined in a manner to resemble how an enterprise does business, e.g. a customer is only allowed to have one account.

The model defines complex relationships between the internal application sources inside the organization, its business and consumer customers, as well as intermediaries and other parties, with the ability to handle complex hierarchies.

The model provides support for managing these hierarchical relationships not only within an authoritative entity but also across authoritative entities. For example, it could model people, organizations, groups, customers, contacts, employees, and suppliers. It models their accounts, locations, classifications, and preferences. Most importantly, it models the vast array of hierarchical and matrix relationships that exist between all the participants in real world operations.

5.6.3 Relationship Management

To support the changing needs of a dynamic business environment Relationship Management caters for complex, real-life relationships between entities. Examples

include charts of accounts, organizational hierarchies, legal entity structures, product hierarchies, customer relationships, sales territories, fiscal calendars, and others.

Understanding and maintaining these relationships are critical for analytical reporting, e.g. enabling reporting rollups needed across global and local regions and lines of business.

Historically, this master data is maintained in multiple systems, which can lead to inconsistencies in reporting and reconciliation issues. Relationship Management allows this information to be defined, maintained, compared, and versioned, centrally.

A relationship represents the way two entities interact with each other, based on the role that each entity takes with respect to the other. Relationship types determine if the relationships created with the type are hierarchical, and if not, whether or not they can be circular.

- **Hierarchical Relationships -** A hierarchical relationship ranks one entity above the other. For example, in an employment relationship, the employer is ranked above the employee. In the employment hierarchy, the employee would be a child that appears below its parent, the employer.
- **Circular Relationships -** If a relationship type allows for circular relationships, you can create a relationship from Party A to Party B to Party C and back to Party A. For example, Party A is a competitor of Party B, which is a competitor of Party C, which is a competitor of Party A.

5.6.4 Entity Consolidation

With entity data typically fragmented and duplicated across operational silos, it is often impossible to determine which version of the entity (in which system) is the most accurate and complete.

Entity Consolidation resolves this issue by acquiring, mapping, and cross-referencing data from multiple source systems while at the same time invoking data quality management capabilities to standardize, cleanse, eliminate duplication, and produce a blended record that takes into account survivorship rules and source system priority rankings for each attribute.

Cross-referencing associates cleansed authoritative records with the corresponding records residing in the operational applications. It maintains this cross-reference even as it eliminates duplicate records as part of the consolidation process. It does this by maintaining the ID of every connected system and maintaining the ID of the object in each connected system with an enterprise-wide unique identifier (EUID).

This consolidation must support both batch and incremental consolidation so that it can support an initial load of entity data as well as provide facilities for loading the entity data incrementally to keep the authoritative from becoming stale.

5.6.5 Entity Lifecycle Management

Entity Lifecycle Management represents the capability to manage the full lifecycle of an entity, from requesting the creation or change of a new record, authoring, augmenting, validating, approving, versioning, and managing the record. This also includes a full set of CRUD (Create, Read, Update, and Delete) operational capabilities, and invoking Data Quality Management.

Supporting the full lifecycle of an authoritative data entity requires access to a flexible and comprehensive workflow mechanism; preferably one that offers a collaborative management approach.

5.6.6 Entity Publication

To ensure that authoritative data management is effective and does not itself become a data silo, the authoritative data must be easily consumed and/or propagated to the needed systems and applications, so that the authoritative data can be fully leveraged.

Entity Publication provides capabilities to share the best version of authoritative data to the operational and analytical applications. Example publication approaches include:

- **Synchronization** - Full or partial synchronization of the central authoritative data with enterprise business processes and the connected applications and insuring that authoritative data stays in sync across the IT landscape.
- **Services** - Maintaining a repository of services to facilitate discovery, and utilization of services to expose authoritative data to consuming applications and business processes.
- **Notifications -** The publication of notifications to interested subscribers by various means (e.g. email, JMS) of either business rule or time driven events. For example - any change to an authoritative entity triggers a business event that in turn could invoke a workflow to act on the event.

5.7 Enterprise Content Management

With enterprises creating and receiving more content than ever before, the creation, management and distribution of this content increasingly becomes a challenge. Enterprise Content Management (ECM) provides the capabilities required to manage and deliver content in the proper format to various information consumers and producers. See Figure 5–18.

Figure 5–18 Enterprise Content Management Capabilities

A unified approach to ECM provides a full array of content management capabilities that an organization requires, such as Document Management, Digital Asset Management, Web Content Management, Information Rights Management, and Image and Process Management.

These diverse content management capabilities leverage repository management capabilities, which enables an organization to turn their unstructured content into assets, and implement a cohesive strategy for securely managing content across their enterprise.

5.7.1 Repository Management

Repository Management provides capabilities that manage all phases of the content life cycle from creation and approval to publishing, searching, expiration, and archiving. All types of content, ranging from e-mail, discussions, documents, reports, spreadsheets and records to images, multimedia or other digital formats, receive the same set of fundamental core services.

5.7.2 Document Management

Businesses generate large amounts of digital content. As content volume grows the discovery and management becomes challenging. Document Management captures, secures, shares, and distributes content to the relevant users.

This allows organizations to save money and improve operational efficiencies by streamlining communications, automating routine tasks, and lowering costs related to the printing, shipping, and storage of business documents.

Document Management leverages repository management capabilities such as automated user-centric processes with document routing and approval, security, and revision control.

5.7.3 Digital Asset Management

Digital Asset Management enables organizations to store, find, and access their brand assets. Digital Asset Management functionality automates the creation of thumbnails and renditions, convert high-resolution assets into Web-friendly formats, and storyboard video to help select key scenes.

Digital Asset Management leverages repository management capabilities such as classification and conversion which defines and provides images and videos in specified formats and sizes to assist organization maintain consistent standards for digital content use.

5.7.4 Web Content Management

Internal and external web sites are key tools utilized by businesses today but development and publishing of content on these web sites can be costly.

Web Content Management allows documents and Web pages to be published as Web sites. It enables accurate, timely, and current Web content with consistent branding and presentation. Content can be updated in minutes based on reusable templates, code, and content.

Web Content Management leverages repository management capabilities such as conversion and publishing where content authored in familiar office document formats are automatically converted to HTML and made part of a fully linked website.

5.7.5 Imaging and Process Management

As previously highlighted, businesses generate large amounts of content. A good portion of this content contains valuable information but remains trapped within paper. Imaging and Process Management provides the capabilities for the storage, classification, and utilization of scanned images of physical business artifacts that are core to the operation of many business functions such as invoices, shipping documents, expense receipts, and other related documents.

The capture of these physical documents is vital as they are easily lost and as volumes increase so does storage costs. Imaging and Process Management enables the capture of physical documents via scanning, the annotation and markup of the images and process oriented capabilities such as routing and approval automation.

Imaging and Process Management also leverages repository management capabilities such as the storage and classification of artifacts and metadata.

5.7.6 Information Rights Management

Information Rights Management secures and tracks sensitive digital information wherever it is stored and used. Encryption is utilized to extend the management of information beyond the repository whether it is used inside or outside of organizations security boundaries.

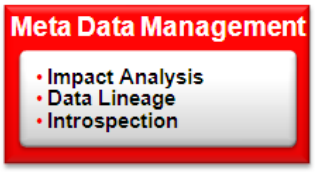
5.8 Metadata Management

Throughout the information management platform, metadata is created and utilized at every level and within every architecture capability.

Metadata can be classified into four classifications:

- **Business Metadata** - Business metadata describes the business meaning of data. It includes business definitions of the objects and metrics, hierarchies, business rules, and aggregation rules.
- **Operational Metadata** - Operational metadata stores information about who accessed what and when. This information is not only important for legal requirements but for the design of the data warehouse itself. For example, we can identify that a particular data mart is not being utilized. This will enable us to develop a plan. Should we eliminate the data mart? Should we be providing better education for the end users? Should we redesign the application or the data mart?
- **Technical Metadata** - Technical Metadata describes the data structures and formats such as table types, data types, indexes, and partitioning method. Also, it describes the location of the data elements. With technical metadata, version control of database structures is possible.
- **Process Metadata** - Process Metadata describes the data input process. It includes data cleansing rules, source target maps, transformation rules, validation rules, and integration rules. Also, it includes the history of data input such as the time of the data loads.

Figure 5–19 Metadata Management Capabilities



As depicted in Figure 5–19, Metadata Management capabilities include: Impact Analysis, Data Lineage, and Introspection.

5.8.1 Introspection

Metadata introspection provides a discovery and harvesting capability whereby objects, attributes, and relationships are discovered, identified, inferred, and populated into a metadata repository. For example, tables within a database, including the contained data, can be analyzed to identify relationships, even where no keys exist. Another example is the harvesting of configuration details across different tool sets.

The populated repository can then be utilized as a means to support impact analysis.

5.8.2 Impact Analysis

Impact analysis provides the ability to see dependencies across the architecture. For instance, when a database table is changed or moved, it helps one understand the effect on other components of the architecture, such as data movement processes, virtualization definitions, and Information Services.

Without impact analysis changes may result in a cascade of failures. This is particularly true when dependencies are not managed by a single technology. Each technology has its own 'understanding' of the architecture, and works on the basis that changes are not being made. Impact analysis requires active introspection, or sharing of knowledge, in a way that provides an end-to-end view of interdependencies.

5.8.3 Data Lineage

Data Lineage, sometimes called "provenance" or "pedigree", is the description of the origins of a piece of data and the process by which it arrived in a database. Lineage is particularly important when multiple copies, versions, or variations of similar data reside across the organization.

In order to ensure the accuracy of information, one must know where the underlying information came from. Without provenance, insight might be derived from stale, incomplete, or inaccurate versions of data. Given the circuitous route that data may travel, the architecture must provide capabilities to understand lineage without reading through or reverse-engineering ETL processes, database procedures, and software programs.

5.9 Data Virtualization

A cornerstone capability in an information provisioning architecture is Data Virtualization. It includes many important functions that enable information to be consumed without direct knowledge of, or connection to, underlying data sources.

Figure 5–20 Data Virtualization Capabilities

Data Virtualization acts as the intermediary and abstraction layer between the information consumers and all physical sources of data that may contribute to the interaction. This may entail transformation, federation, caching, and connectivity to heterogeneous data sources.

5.9.1 Federation

Despite best efforts by organizations to consolidate information, many find themselves with multiple, distributed data stores. These organizations require an approach of accessing these distributed data sources as if they were a single data store. Using

federation, information consumers such as applications and users can access and modify information at multiple data stores as if it resided in a single data store.

Federation is a metadata driven approach (a.k.a Enterprise Information Integration - EII) that provides the ability to combine and aggregate information from multiple data stores into a real-time view within a single operation. Federation generally comes in two forms:

- Data Federation
- Content Federation

5.9.1.1 Data Federation

Data Federation provides a logical representation of a data entity that supports access simplification for information consumers that need to access data across multiple data stores. Two common approaches to achieving data federation are:

- **SQL Gateway** - The approach utilizes a database that provides a gateway to other databases and provides a virtual view. Data appears to be integrated in a single virtual database, while actually remaining in its current distributed locations. Access is via the database but it is federated because the database fans out the request to other databases. This approach is useful when organizations want to perform ad hoc queries or updates on infrequently accessed data that is more appropriately located elsewhere.
- **Federated Data Services** - The second approach utilizes Federated Data Services which enable access of disparate data stores via a services interface. The Federated Data Services resolve any semantic conflicts by supporting a model driven approach to mapping and transformation. This approach is appropriate when there are few joins and small result-sets.

5.9.1.2 Content Federation

Content federation utilizes federated queries of local and remote content repositories simultaneously utilizing standardized APIs from a single point of access (SPOA). The results of the federated queries are aggregated and returned according to the specified criteria, e.g. results are ordered by date of creation. These queries encompass both textual and metadata searches.

5.9.2 Caching

Scalability and bottleneck issues can arise when many consumers (e.g. tools, services) access data simultaneously which may overwhelm the individual data sources. For example, data sourced from legacy systems may not be architected in a way to support the data access demands that result from wider usage.

One common approach to address these needs is the use of caching. Caching provides the building blocks necessary to duplicate data in memory so that future data access requests can be served faster. Caching provides several benefits including scalability and performance. In a distributed architecture caching is more complex as data needs to be kept synchronized and consistent.

Data Virtualization through a data grid addresses these challenges by providing an in-memory replicated and distributed data caching service ensuring reliability and high availability. In effect a data grid is used to support a cluster of objects, (e.g. java), that are in turn mapped down to a data store. The data grid coordinates updates to the data by using cluster-wide concurrency control, replicating and distributing data modifications across the cluster, and delivering notifications of data modifications to any servers that request them.

The data grid capability is good for near real-time access to java objects, when it is not feasible in terms of performance to go to the source data. For instance, as a buffer against a legacy data store, a data grid can serve as a caching layer that scales linearly to extreme levels. IT does not have to re-architect the legacy system; the data grid enables IT to offer broader access with higher service levels.

For more details regarding caching refer to the *ORA Application Infrastructure* document.

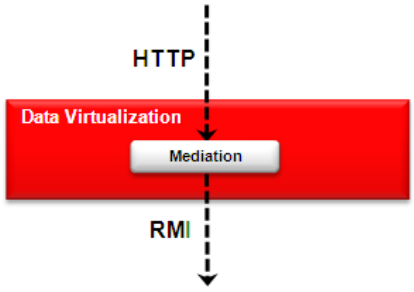
5.9.3 Mediation

Mediation is a key capability to enable loose coupling between information consumers and information providers. Mediation can be broadly defined as resolving the differences between an information consumer and an information provider in order to integrate them seamlessly. Typically an enterprise has a variety of information consumers and providers that are fundamentally different. Rather than embedding mediation logic into the information consumer and provider, the mediation capability is provided by the Information Provisioning Layer as part of data virtualization.

In order to be flexible and allow for a wide variety of heterogeneous service invocation techniques, data virtualization supports transport mediation, message exchange pattern mediation, and security mediation.

- **Transport Mediation -** Allows the information consumer to invoke an information service using a different transport than that supported by the provider. Data Virtualization achieves this by translating the transport protocol of various communication protocols such as HTTP, file, JMS and FTP. See Figure 5–21.

Figure 5–21 Transport Mediation



- **Message Exchange Pattern (MEP) Mediation** - MEP mediation resolves the difference between the invocation patterns of the information consumer and information provider. The most common MEP used is the synchronous request/response pattern. Data Virtualization can mediation different MEPs using Sync-to-Async or Async-to-Sync bridging.
- **Security Mediation -** resolves differences in security tokens, identities, attributes, etc. in order to enable secure communication in spite of technology differences between the consumer and provider.
- **Message Mediation** - Message mediation manipulates and transforms messages as they travel from an information consumer to information provider and optionally back to the information consumer. Message mediation might include aggregation, enrichment, filtering, and wrapping. Message mediation manipulates and transforms messages as they travel from an information consumer to information provider and optionally back to the information consumer. Message mediation might include aggregation, enrichment, filtering, and wrapping. While

this capability is not meant to fully address all canonical message transformation requirements, it does give an architect the flexibility to abstract current and future differences in how data are stored to how data is required to be consumed. This allows for Information Services to be utilized for a wider audience, by being able to adapt the input and output to an information service invocation.

Refer to *ORA SOA Infrastructure* and *ORA Integration* documents for more details on mediation. In addition refer to *ORA Security* for details on security mediation.

Refer to *ORA SOA Infrastructure* document for more details.

5.9.4 Routing

Loose coupling is the ability to decouple the information consumer from the implementation specifics of the information provider. Routing refers to the ability to control where an information consumer request is sent, or which information service is invoked. This can be accomplished by extracting data that affects routing decisions from the contents of the request message (content based routing), message header, or via configuration data (configuration based routing).

Refer to *ORA SOA Infrastructure* for more details.

5.9.5 Policy Enforcement

Rules that define acceptable access to information can be described in a policy. Policies may include regulatory compliance rules, business policies, security policies, SLA policies, and validation rules. Data virtualization should be able to support the programmatic enforcement of a variety of policy types.

To support the principle of abstraction, the Data Virtualization Layer supports the enforcement of policies independent of the implementation of the information consumer. Policy enforcement can happen at different enforcement points based on the use case.

Policy enforcement points (PEP) can be applied in many forms but it is common to utilize either a gateway or agent approach that intercepts requests and/or responses and enforces the policies that are attached to the requests and responses. For example - routing and prioritization of an information request based on business criteria, and deciding whether an information consumer has authorization to access the information.

Refer to *ORA Management & Monitoring, ORA SOA Infrastructure, and ORA Security* documents for more details.

5.9.6 Search

Search provides the ability to search and locate data across intranets, databases, (distributed) file systems, email, document repositories, applications, and portals, utilizing textual and/or metadata keywords.

Search provides advanced full text search across any single or combination of fields, leveraging a variety of configurable term matching and relevancy ranking algorithms to retrieve results for display and analysis purposes. In addition, this capability supports faceted search. Faceted search (aka guided navigation) accesses information organized according to a faceted classification system for both structured and non-structured data, enabling users to search a collection of information by applying multiple filters

When searching internal content it is imperative that it is done in a secure manner to make sure that search results and content retrieval does not return personal and/or confidential content.

5.9.7 Connectivity

Connectivity provides a number of mechanisms to access data sources and provide a level of encapsulation to hide the complexity of connecting to data sources. This enables other capabilities to treat data sources in a more uniform and generic fashion.

Generally, both synchronous and asynchronous techniques are employed by capability to provide connectivity. Both proprietary and standards-based technologies are used. The key mechanisms within this capability include:

- **Messaging** - Messaging provides asynchronous connectivity. Many traditional integration products are based on messaging systems.
- **Adapters** - Adapters provide a standardized approach for connecting to applications.
- **Standard APIs** - Applications may provide access via one or more standard application programming interface (e.g. JDBC, RMI, WS*).
- **Custom APIs** - Some applications (especially legacy) only provide access via a custom application programming interface. The connectivity layer provides the capability to call these custom APIs and wrap them with a standardized interface.
- **Events** - Events allow the applications and data sources to initiate actions.
- **Connection Pools** - Connection pools are caches of database connections (e.g. JDBC) maintained so that the connections can be reused when future requests to the database are required.

Refer to *ORA Integration* document for more details.

5.10 Information Access

The Information Access Layer should be seen as the single point of information access regardless of transport and access mechanism. This allows many types of information consumers the flexibility to access Information Services and information in a variety of protocols and APIs. See Figure 5–22, "Information Access"

Figure 5–22 Information Access

Information is accessed using standards-based protocols and APIs, such as Open Database Connectivity (ODBC), Java Database Connectivity (JDBC), and Web Services interfaces (WS*, REST). This enables the flexibility of many types of information consumers to access and make use of information and Information Services.

5.11 Information Services

In order to best provide and advertize information to a diverse range of consumers, it is advantageous to deliver Information Services. Information Services logically abstract all underlying architectural concerns, e.g. provisioning, virtualization, etc.,

and present information via a clearly defined interface. Therefore Information Services are used to access information from various sources using many different technologies, and present information in a business-friendly form. This creates an abstraction between the consumers of information and the sources of information. Consumers of information do not need to be concerned with how it is stored, or how it is represented in its native form. They can work with representations (schemas) of greater business value, such as canonical data models.

Information Services can be virtualized, aggregated views built from sources across the enterprise. They simplify information access and updates, and once created, are highly reusable. Information Services expose highly optimized engines for working on all types and formats of information. Information may originate in various data stores such as databases, flat files, XML files, and legacy systems, or may originate within the Information Provisioning Layer where it may have been aggregated, transformed, and synchronized.

A fully-defined information management architecture will include many types of Information Services. Fig Figure 5–23, "Example Information Services" below shows various classifications of Information Services. By no means are these the only functional classifications for Information Services. The collection of aforementioned Information Service profiles is meant to give guidance to an architect when planning a multi-year information management rollout strategy that might include a range of different Information Services for different kinds of use cases.

Figure 5–23 Example Information Services

5.11.1 Data Services

Data Services provide a single point of access for aggregated, real-time, operational and historical data. Data Services operate on, and expose, a common business information model. This is a logical semantic data model that is backed by one or more physical data sources. The purpose of this model is to provide consumers with a clear semantic view of data irrespective of physical data model characteristics.

Data Services are aware of constructs such as data relationships and objects. This allows Data Services to be as simple as fetching data from a database, fetching data from a data grid or via federated data source queries in real time with aggregated data result sets. In addition, Data Services can enable transactional create, update, or delete operations to an underlying data store while maintaining business and referential integrity rules.

These Data Services also leverage the virtualization capabilities of the information management platform in order to perform the relevant mediation, i.e. logical to physical mapping. This allows consumers to be isolated from any future changes to the underlying source systems. Changes should be handled by simply updating the metadata within the virtualization layer.

5.11.2 Analytical Query Services

Analytical Query Services are an extension of Data Services, designed specifically for analytics. In addition to virtualization features they are aware of dimensional constructs such as hierarchies and aggregations. This combination of features enables

them to satisfy queries based on data from multiple physical sources, either standard normalized (via SQL) or multi-dimensional (via MDX). Analytical Query Services are also "aggregate-aware", which means that queries that involve aggregations or calculations can use tables of pre-aggregated values if they exist, or perform aggregations and calculations "on the fly".

Analytical Query Services can support operations that go beyond the realm of standard SQL queries. For example, to determine market share changes versus a year ago, a service may need to perform functions that involve row to row comparisons, time-based reporting, and derived measures (ranks, Ntiles, standard deviations, moving averages, etc.). Complex business measures such as these add value in terms of capability, efficiency, and performance, over standard Data Services.

Refer to *ORA Business Analytics* documents for more information on Analytical Query Services.

5.11.3 Content Services

Content Services operate on and expose content irrespective of format or physical location, whether it is stored in a content management repository or a file store.

Content Services enables content search and discovery capabilities such as free text search or search of metadata associated and/or contained within the artifact in a repository. As well as searching, Content Services enable the retrieval of specific artifacts from a repository based upon a unique identifier, such as a URL.

In addition to requesting individual artifacts directly, Content Services enable a form of subscription where an information consumer is automatically notified when new documents are added to a repository in accordance with a predetermined policy or profile.

5.11.4 Spatial Services

Spatial Services are designed specifically for geospatial data. They provide the ability to work with maps, points, lines, and polygons. Consumers can create, query, update, and delete features within a geospatial map.

5.12 Provisioning Services

Provisioning Services are services that act solely on the capabilities within the Information Provisioning Layer, unlike Information Services which focus primarily on the consumption of information. In effect Provisioning Services are services that operate on information management that expose highly optimized engines for the provisioning and distribution of data. Therefore the underlying architectural concerns of the Information Provisioning Layer have been abstracted and presented in a clearly defined interface.

A fully defined information management architecture will include many types of Provisioning Services. Figure 5–24, "Provisioning Services" below shows various classifications of Provisioning Services, which are meant to give guidance to an architect when planning a multi-year information management rollout strategy that might include a range of different Provisioning Services for different kinds of use cases.

Figure 5–24 Provisioning Services

5.12.1 Data Ingestion Services

Data Ingestion Services offer and leverage data movement and data processing capabilities to distribute and replicate various volumes of data in both batch and real-time mode via a uniformed standards-based interface.

For example a bulk data ingestion service which invokes an ELT-style job could be invoked as part of a business process.

5.12.2 Analytical Services

Analytical Services offer and leverage analytical capabilities within the Information Provisioning Layer that are presented in a usable format. This enables a programmatic approach to execute analytic functionality such as creating and deleting dimensions.

5.12.3 Data Quality Services

Traditionally Data Quality has been applied in batches to clean up large data stores but now the usage is shifting to include a real-time and preventative use case, to cleanse the data before it's a problem.

Data Quality Services offer and leverage data quality capabilities to clean up various types of payloads and present them in a usable fashion. Data Quality Services can come in various profiles. For example:

- **In-line -** Data Quality Services are used in-line with other Information Provisioning capabilities such as Data Movement., i.e. Bulk Data Movement via ETL. These services help in cleansing and validating data as it is being moved and integrated. This gives the flexibility to ensure that the data are always consistent before it is moved into the target.
- **Static** - These Data Quality Services invoke steps such as data cleansing and standardization on static data stores (e.g. to clean up a legacy database).

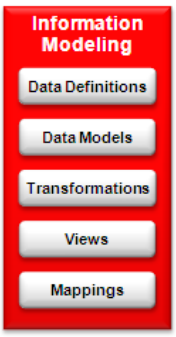
5.12.4 Repository Services

Repository Services offer repository capabilities within the Information Provisioning Layer that are presented in a usable format. This enables a programmatic approach to execute capabilities found within repositories (e.g. content management repositories, metadata repositories), such as notification and workflow.

5.13 Information Modeling

Information Modeling is a cross-cutting concern as it can be applied to many capabilities across all layers and components of the information management architecture.

Figure 5–25 Information Modeling Capabilities



A multi-layered approach to modeling (conceptual, logical, physical) supports the definition and deployment of a common logical information model and associated semantics, which enables information sharing and consistency. In conjunction with metadata it can also support and enable a dynamic and declarative environment by defining models such as transformation models, source to target mappings, and various process models.

Models are not only used to organize information, but they are also used across the architecture for introspection and impact analysis.

Table 5–2 describes a number of ways that modeling can be applied across the information layer.

Table 5–2 Example Information Modeling Constructs

Architecture Component	Modeling Constructs
Data Movement	Data Movement Processes, Transformations, Mappings, Metadata
Data Governance	▪ Data Definitions & Profiles ▪ Relationships ▪ Data Quality Processes ▪ Rules ▪ Mappings ▪ Transformations ▪ Metadata
Historical Data Management	▪ Data Definitions ▪ Schemas ▪ Relationships ▪ Views ▪ Metadata
Analytical Data Management	▪ Data Definitions ▪ Schemas ▪ Aggregations ▪ Views ▪ Cubes ▪ Metadata

Table 5–2 (Cont.) Example Information Modeling Constructs

Architecture Component	Modeling Constructs
Authoritative Data Management	■ Data Definitions ■ Lifecycle Process ■ Schemas ■ Relationships ■ Mappings ■ Metadata
Enterprise Content Management	■ Taxonomy ■ Workflows ■ Metadata
Data Virtualization	■ Common Information Model ■ Transformations ■ Mappings
Information Services	■ Service Models (Contracts, Data Definitions, Relationships) ■ Metadata
Information Access	■ Interfaces

6

Logical View

The information management Logical View is comprised of several scenarios that pertain to specific aspects of information management. The purpose is to drill down into the architecture with a certain context in mind, and provide greater detail than would otherwise be feasible on a single universal architecture diagram.

A fully defined information management architecture will include many types of information management scenarios. These scenarios are grouped into three distinct areas.

- Information Consumption
  - Virtualization
  - Information as a Service
- Information Acqusition
  - Ingestion
  - Replication
- Information Organization
  - Big Data Processing
  - Enterprise Master Data Management
  - Data Warehousing
  - Information Provisioning as a Service

The collection of aforementioned information management scenarios is meant to give guidance to an architect when planning a multi-year information management rollout strategy that might include a range of capabilities for different kinds of scenarios.

6.1 Information Consumption

This section details two example information consumption scenarios, virtualization, and information as a service.

6.1.1 Virtualization

Virtualization allows organizations to leave data in its existing location, rather than moving it to a central repository and acts as the intermediary and abstraction layer between the information consumers and the sources of data that may contribute to the interaction. This may encompass federation, Service Oriented Integration (SOI), and Enterprise Application Integration (EAI). Virtualization leaves the data in place without requiring copying or synchronization logic.

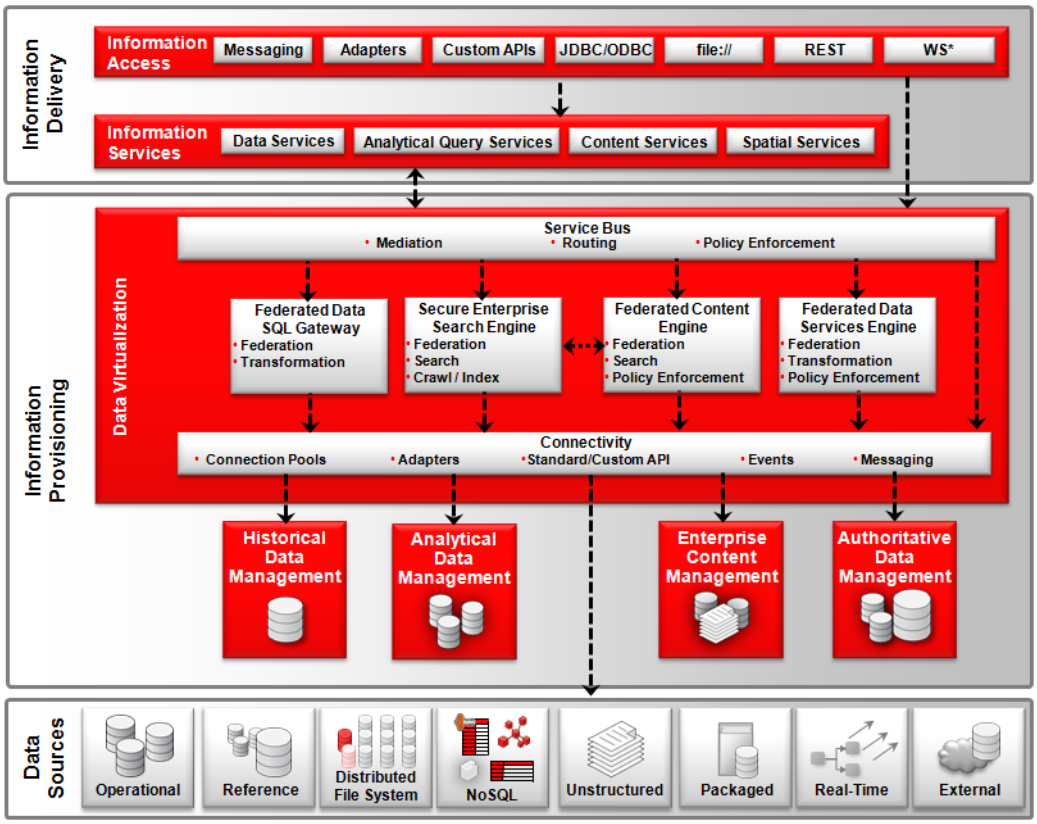
The table below highlights some of the key characteristics that virtualization offers.

Table 6–1 Virtualization

Approach	Characteristics	Challenges
Virtualization	■ Supports real-time access to data source systems by leveraging various techniques such as federation, SOI, EAI, which assists in limiting the number of copies of the data in the organzation. ■ Supports real-time access to data that chnages rapidly and frequently. ■ Supports basic semantic transformations. ■ Supports simple and few joins. ■ Supports small result sets. ■ Quick Deployment. ■ May support real-time update across source systems. ■ Not suited for large data replication.	■ Reliant on performance of source systems and virtualization infrastructure. ■ Reliant on availability of source systems. ■ Limited or no data quality facilities, therefore primarily reliant on data quality of source systems. ■ Transaction integrity.

Figure 6–1, "Virtualization - High-level logical diagram" below expands on these characteristics and highlights the interactions between key components to support virtualization.

Figure 6–1 Virtualization - High-level logical diagram



6.1.1.1 Information Access & Services

The Information Provisioning Layer has two primary entry points:

- **Information Access** - Allows many types of information consumers the flexibility to access the Information Provisioning Layer in a variety protocols and APIs. For example tools which do not support services connectivity can still access the Information Provisioning Layers using standard supported protocols such as ODBC ad JDBC.
- **Information Services** - Allows access to the Information Provisioning Layer and information via a clearly defined interface. Therefore Information Services are used to access information from various sources using many different technologies, and present information in a business-friendly form.

All requests, whether they are from the Information Access Layer or Information Services Layer, are funneled through the Data Virtualization Layer. This layer provides several capabilities, most of which are optional. For instance, federation, aggregation, and transformation will not be required for all queries. However, abstraction is highly recommended as a means to avoid tight coupling between data consumers and providers.

6.1.1.2 Data Virtualization

The Information Provisioning Layer supports a service bus that handles key capabilities such as mediation, routing, and policy enforcement.

In addition the Information Provisioning layer includes four optional virtualization engines, as shown in Figure 6–1, each having their own characteristics and usage:

- **Federated Data SQL Gateway** - Provides a virtual view which provides access to remote data while translating SQL dialects, data types, and data dictionary. This engine supports an ODBC/JDBC interface.
- **Federated Data Services Engine** - Provides a virtual view by resolving any semantic conflicts by supporting a model-driven approach to mapping and transformation. This engine supports both a services interface as well as an ODBC/JDBC interface.
- **Federated Content Engine** - Provides a virtual view to accessing content from disparate content stores utilizing standardized APIs.
- **Secure Enterprise Search Engine** - The secure enterprise search engine provides virtual access to pre-crawled content in local and remote enterprise search engines.

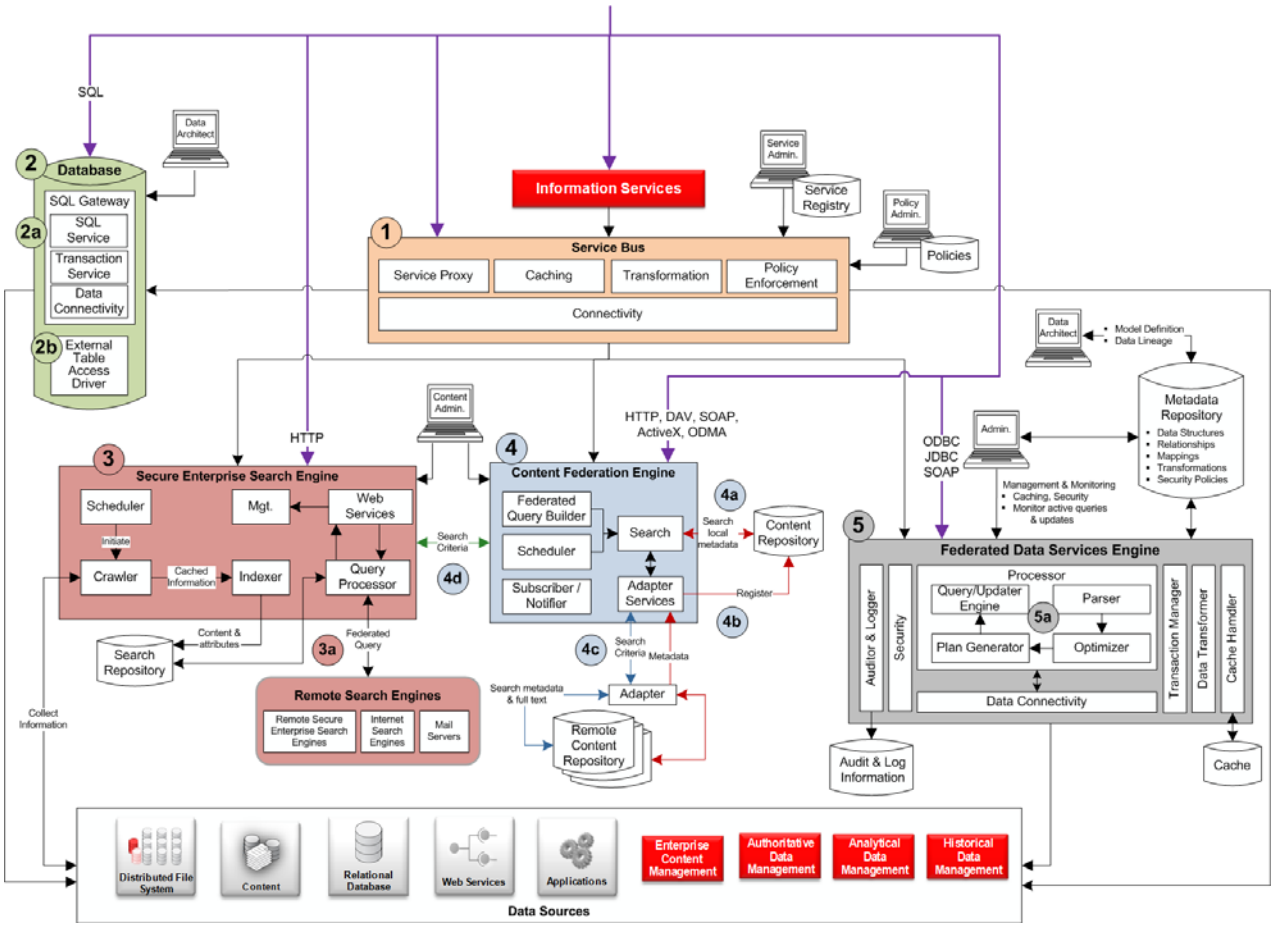
6.1.1.3 Connectivity

The connectivity layer provides a number of synchronous/asynchronous mechanisms and pool mechanisms to access information and data sources. This provides a level of encapsulation to hide the complexity of connecting to data and information sources. Connectivity will often be physically built into the other data virtualization components, as opposed to being a standalone component of the architecture. This layer can also encapsulate an SOI or EAI infrastructure. For more details regarding SOI and EAI refer to the *ORA integration* document.

6.1.1.4 Interactions

More detailed interactions can be seen in Figure 6–2, "Virtualization - Interactions" below.

Figure 6–2 Virtualization - Interactions



As noted earlier in this section there are two main points of entry (Information Access and Information Services). For clarity, information access entry points are shown in purple and access one of the five main components. The Information Services can access the same main components via the service bus.

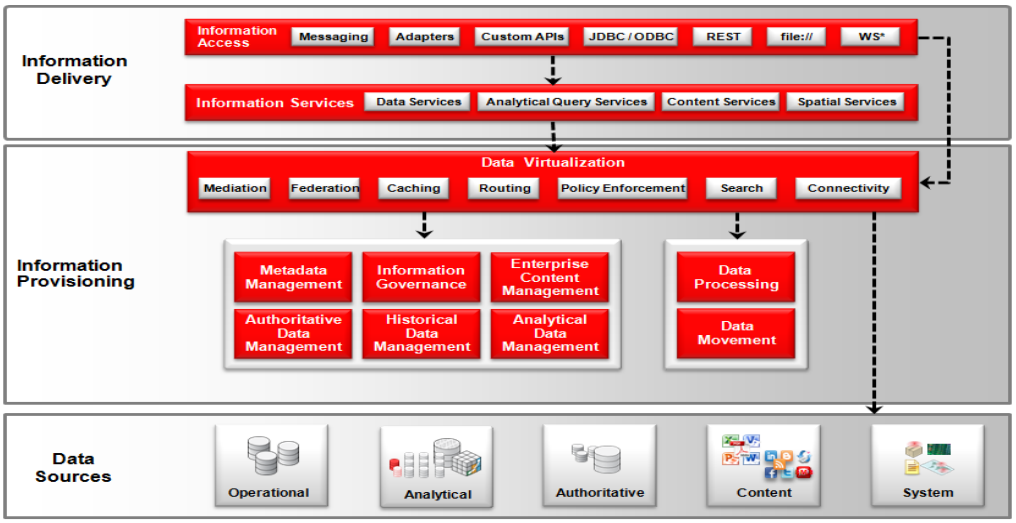
1. **Service Bus** - The service bus provides mediation, routing and policy enforcement capabilities. The request is then forward to the appropriate engine.
2. **Relational Database** - A relational database provides capabilities that abstract away remote connectivity to both relational and flat file data.
    a. **SQL Gateway** - The SQL gateway resembles a data hub because accessing the data is performed by the database but it is federated because the database fans out the request to other databases. The database utilizes an embedded SQL gateway to access remote data while translating SQL dialects, data types, and data dictionary. While access is achieved via an SQL interface it is common for this interface to be further abstracted by the service bus to enable a service interface to the SQL gateway.
    b. **External Tables** - External Tables enables access to data/files as if they were tables in the database. An external table is defined in the database enabling direct SQL access to data files stored on either a file system or a distributed file system. Not only can the data in the external tables be queried by SQL, it can also be joined with data stored in database. This approach has the added advantage of enabling organizations to utilize their existing skillset to access disparate data.

3. **Secure Enterprise Search Engine** - The secure enterprise search engine queries its search index repository to find appropriate matches. The search index is populated via a scheduled crawl of remote content.
    a. The secure enterprise search engine can be configured to perform a federated query against remote enterprise search engines, mail servers and internet search engines.
4. **Content Federation Engine** - The content federation engine can utilize a number of mechanisms to support content federation such as Catalog Search, Remote Repository Search, and Search Engine. This enables a virtual repository which provides an enterprise view of all content assets regardless of where they are stored in the enterprise. This virtual view provides a single point of entry for access and to obtain fully aggregated search results from separate content sources.
    a. The content federation engine performs a catalog search by querying the metadata of the central content repository for matching content. This approach tends to be the fastest search option.
    b. Using the content federation engine performs a catalog search by querying the metadata of the central content repository for matching content. This approach tends to be the fastest search option.
    c. The remote repository search searches all remote content stores. Because of the unknown latency of the search results, this approach tends to be performed in a batch mode. This approach provides the most accurate and up to date results but it may take a long time to complete.
    d. The content federation engine can re-direct the request to a secure enterprise search engine which has already performed a scheduled crawl of content. This allows for content which the local and remote content repositories are unaware of to be included in the search.
5. **Federated Data Services Engine** - Data Architects model and map multiple physical data sources into a set of services defined by logical entities rather than physical location. In addition to defining the entity relationships, data architects define any required transformations.
    a. The request is parsed and optimized to produce an execution plan that utilizes the specific capabilities of the underlying databases. The Federated Data Services Engine can support full CRUD transactions (Create, Read, Update, and Delete) of data in place in the original sources.

6.1.2 Information as a Service

Information as a Service is a service-oriented approach to consuming information in a consistent and secure manner (See Figure 6–3, "Information as a Service" below).

Figure 6–3 Information as a Service



Via the implementation of Information Services, Information as a Service logically abstracts all underlying architectural concerns, (e.g. virtualization), and presents both operational and analytical information via a clearly defined, pre-integrated, and simplified interface.

Information Services can be virtualized, aggregated views built from sources across the enterprise. They simplify information access and updates, and once created, are highly reusable. Information Services expose highly optimized engines for working on all types and formats of information. Information may originate in various data stores such as databases, flat files, XML files, packaged applications, and legacy systems, or may originate within the Information Provisioning Layer where it may have been aggregated, transformed, and synchronized.

Information Services are used to access information from various sources using many different technologies and present information in a business-friendly form. This creates an abstraction between the consumers of information and the sources of information. Consumers of information do not need to be concerned with how it is stored, or how it is represented in its native form. They can work with representations (schemas) of greater business value, such as canonical data models.

Information Services do not include any business logic. This allows them to be reused in different business contexts without modification. They can represent business objects, such as customer, employee, and order, with the associated logic for ensuring consistency of such constructs. Information Services can also provide enhanced features, such as caching and redaction. Redaction is the ability to reduce the information returned by a service according to the entitlements of the service consumer.

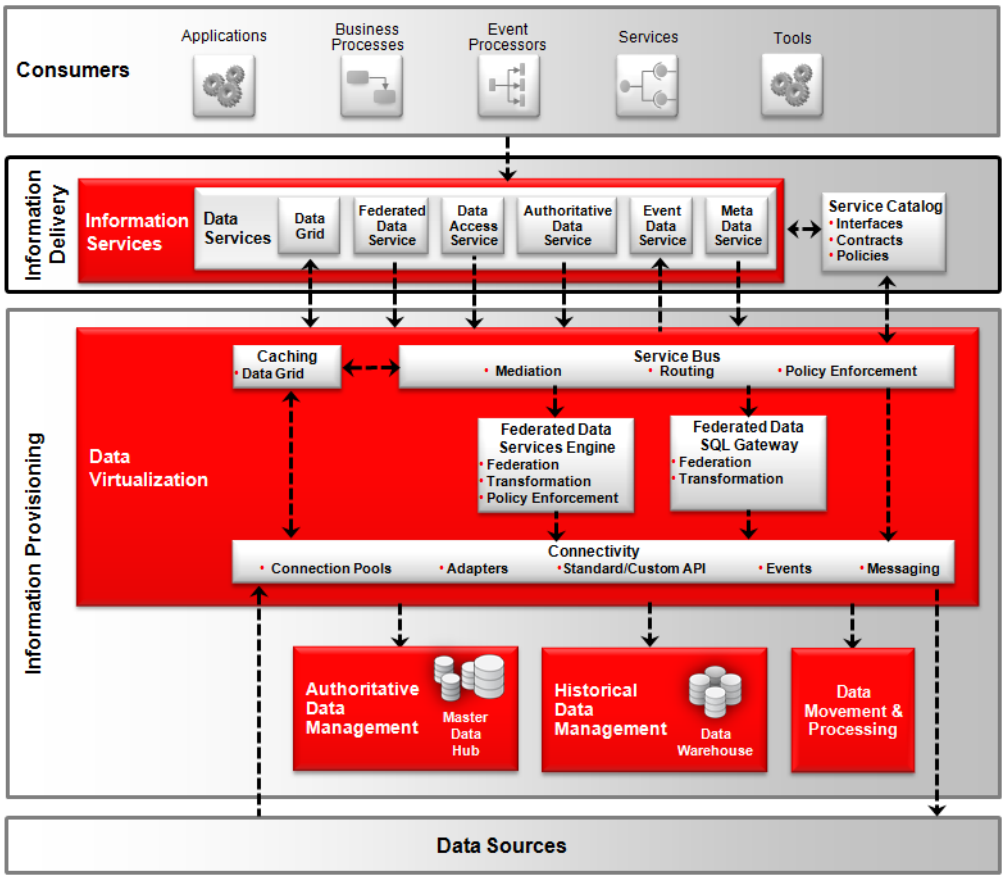
Information Services expose control points for information that are easy to access, publish, and discover. Information Services should be designed to meet the specifications of enterprise-class SOA Services. This level of robustness enables them to be discovered and reused across the organization, and provides the level of management and autonomy needed to support load requirements and quality of service agreements. See *ORA SOA Foundation* for more details.

As highlighted in section Section 5.11 within the Conceptual View chapter, a fully defined information management architecture will include many different classifications of Information Services. Here are some example classifications of Information Services.

- Data Services
- Analytical Query Services
- Content Services
- Spatial Services

To expand on these concepts Figure 6–4 below shows some examples of Data Services. The collection of Data Services shown in the diagram are meant to give guidance to an architect when planning a multi-year information management rollout strategy that might include a range of different Data Services for different kinds of use cases.

Figure 6–4 Example Data Services



Data Services represent queries that have been pre-configured for a specific purpose. The request and response of these services is formalized in a contract and/or interface specification. They offer the benefits of being pre-engineered (tested, documented, and governed), to standardize results and maximize the efficiency and performance of common, known queries. Here are some examples of Data Services:

- **Data Access Services** - Data Access Services are the foundational basic Data Services that have been popularized by SOA, that enable consumers of data to not be concerned with where data are stored, how it is stored, or how it is represented in its native form. These services focus on accessing data from various sources using many different technologies, and presenting data in a business-friendly form via a simplified query interface. Data may originate in various databases, flat files, XML files, and legacy systems.

- **Federated Data Services** - Federated Data Services aggregate data from multiple sources into a single service interface view. These services resolve any semantic conflicts by applying the appropriate mapping and transformation activities.
- **Authoritative Data Services** - Authoritative Data Services provide full lifecycle management for business entity data to enable the ability to build or transform existing operational systems into MDM Aware consumers. (See Section "MDM-Aware Consumers" for more details). Through match and fetch data service requests between operational systems and MDM hubs, these services provide access to the single source of truth for authoritative data, thus preventing duplicate and inaccurate data entry.
- **Data Event Services** - Events that occur on business data are captured, correlated, and propagated to Data Events Services so that consumers of these events can take the appropriate action. Events may originate either within the Information Provisioning Layer (e.g. via CDC) or within a data store (e.g. via database triggers). Consult the *ORA EDA documents* for more details regarding event driven architectures.
- **Data Grid Services** - Data Grid Services provide exceptionally fast caching for data access. Very fast, in-memory data frequently needs to span multiple applications, due to geographical factors, or to overcome the limitations of RAM capacity on a single host. Data grid services are not a replacement for persistence; they are typically used in combination with relational databases for storing the data and for maintaining accurate lifecycle controls on the data.
- **Metadata Services** - Metadata Services provide full lifecycle access to metadata object definitions within repository stores. This allows consumers to access object definitions such as schemas, transformation maps, and models.

In addition to the example data services highlighted above, Data Services can also be formed via composition. Composite Data Services are comprised of other Data Services and are hence, Service Consumers; since they provide Services they are also Service Providers.

Data Services are cataloged in a way that supports discovery and promotes reuse. The interface specifications, usage policies, access policies, etc., are maintained in a service catalog. These artifacts affect the definition of services (Information Services Layer) as well as the implementation (Data Virtualization layer).

All Data Service requests are directed through the Data Virtualization layer. This layer provides several capabilities, most of which are optional and will not be required for all queries. However, abstraction is highly recommended as a means to avoid tight coupling between data consumers and providers. The Data Virtualization Layer can route the request to one of several locations depending on the Data Service request. For example Federated Data Service requests can be routed to one of two Federated engines, Authoritative Data Service requests can be routed to one or more master data hubs, Data Access Service requests can be routed to either operational stores or relational data within a Data Warehouse.

6.2 Information Acquistion

There is no one way to acquire information and it is up to the architect to decide which information acquisition technique(s) meet their requirements. Each technique has its own appropriate usage in an information management strategy. See Table 6–6 for characteristics and challenges for each information acquisition technique.

Table 6–2 Ingestion, and Replication/Synchronization characteristics

Approach	Characteristics	Challenges
Ingestion	■ Supports access to both operational & analytical ingestion (e.g. migration, consolidation). ■ Supports various topologies, latencies, communucation styles, and data formats. ■ Supports complex transformation and joins. ■ Supports large results sets. ■ Supports the provision of quality data. ■ Supports high query performance needs. ■ Supports central authoring of authoritative data. ■ Supports analytics.	■ Timeliness of data (Dependent on time window). ■ Business case restriction to copying data. ■ Data synchronization via incremental updates. ■ Data remediation (two-way Synchronization). ■ Implementation Complexity. ■ Generation/mapping of unique identifiers. ■ Definition of a Canonical Model. ■ Propogate source system security requirements into target data store. ■ Source and target data model updates
Replication & Synchronization	■ Primarily for read-only - offload reporting from production source system. ■ Supports both synchronous and asynchronous replication. ■ Supports high performance replication. ■ Supports real-time and near-real time requirements. ■ Supports HA/DR of source systems.	■ Data Sprawl and associated management and control Issues. ■ Data synchronization. ■ Incremental updates. ■ Reliant on data quality of source systems.

A comprehensive information management strategy will utilize all of the above information acquisition techniques and not solely rely on one approach for all circumstances.

6.2.1 Ingestion

Ingestion is an approach whereby data needs to be captured, copied, or moved to different target sources/environments with minimal impact on source systems and query workloads.

This requires the handling of a variety of data (e.g. structured, semi/un-structured), with a range of latencies, utilizing differing persistence models whilst combining architecture capabilities from various technology strategies (Information Management, EDA, SOA).

There are two primary uses of ingestion, analytical ingestion and operational ingestion.

- **Analytical Ingestion** - Focuses on ingesting data from various sources for the purposes of performing analytics. This requires ingestion of data from various sources into a consolidated hub such as Dataw Warehouse, Master Data Management, or for downstream processing such as Big Data Processing.

- **Operational Ingestion** - Focuses on the needs of operational applications and databases. This enables operational data to be captured, filtered, transformed, and potentially consolidated from various operational systems into a single data model.

6.2.1.1 Communication Style

The communication style employed when ingesting data from a source data store can be characterized as either a push or pull technique.

The push technique is reliant on the source data store to initiate the push of data when a change occurs.. Whereas, a pull technique relies on data being pulled from a source data store (this is generally less intrusive). The pull technique often requires a mechanism to identify what has changed since the last pull. In addition the pull technique may need to be scheduled at a time when impact on operational performance will not be a problem.

6.2.1.2 Ingestion Latency

The need and timeliness of data should be taken into account when deciding which ingestion mechanism to employ. This comes down to the degree of latency that is acceptable for a user from the data being ingestion from the source system to the target systems. Table 6–3 below highlights some common approaches.

Table 6–3 Ingestion Latency

Ingestion Latency Approach	Description
Batch	This is still the most common approach to ingestion whereby data is ingested on a regular interval, e.g. monthly, weeky, daily
Micro-Batch	This approach ingests the delta of the data since the last data ingestion. Data is ingested on a more frequent basis then in a batch mode, e.g. every 5 minutes.
Trickle-Feed	This approach ingests data as soon as it becomes available. Data is continuously being ingested

6.2.1.3 Ingestion Topologies

There are a number of ingestion topologies that are commonly employed to address organization's ingestion requirements. See Figure 6–5 below.

Figure 6–5 Common Ingestion Topologies

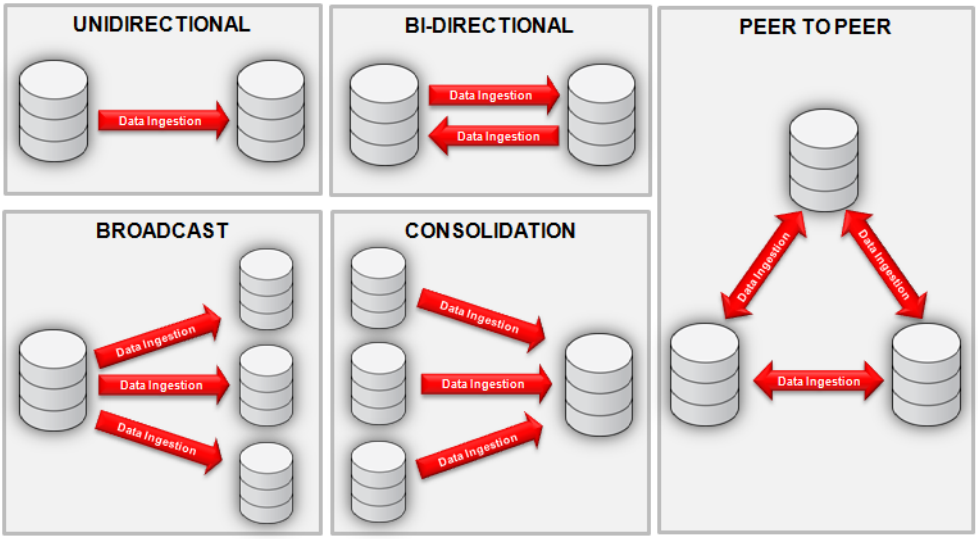


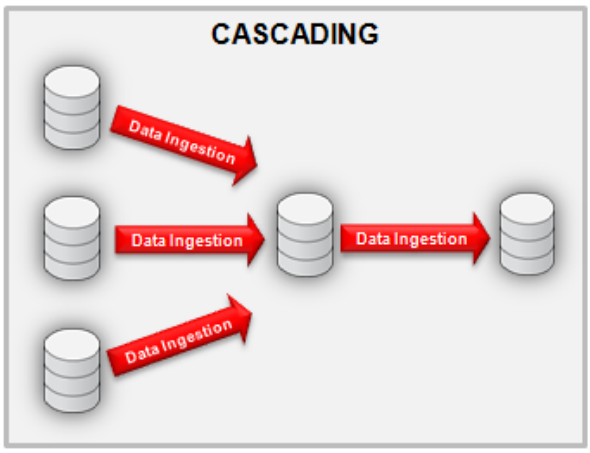
Table 6–4 below provides a description of these topologies and some sample use cases.

Table 6–4 Ingestion Topologies

Ingestion Topologies	Description
Unidirectional	Ingests data from one source data store to a target data store. Example uses cases include offloading reporting requests from production systems and live standby for failover purposes.
Broadcast	Similar to unidirectional pattern but used for ingestion of data to several target data stores. An example use case includes data distribution to several databases which can be utilized for different and distinct purposes, i.e. reporting, test environment, etc.
Bi-directional	Ingests changed data bi-directionally between two data stores to keep both sets of data stores current. Example use case includes an active-active instant failover.
Peer to Peer	Similar to the bi-directional pattern but data is kept current in multiple databases forming a multi-master configuration. Example use cases include load balancing as load is shared between the databases equally, and high availability as additional databases will still be available if one database fails.
Consolidation	Ingests data from several source data stores to a single target data store for the purposes of data consolidation.

While this is not an exhaustive list of ingestion topologies these foundational ingestion topologies enable an architect to define additional more complex topologies by combining two or more topologies together. For example see Figure 6–6 below, a cascading ingestion topology can be obtained by combining the consolidation and unidirectional ingestion topologies.

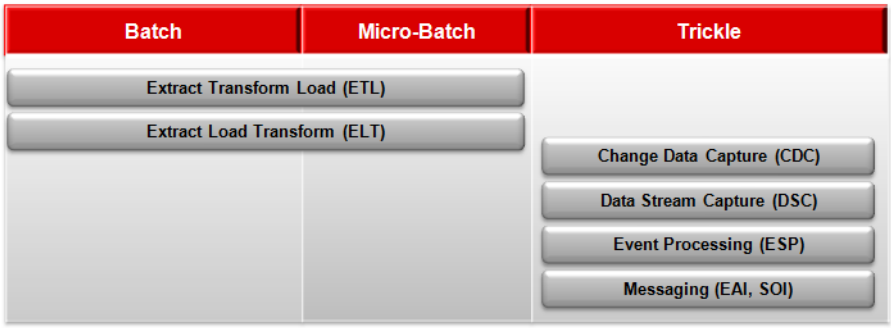
Figure 6–6 Cascading Ingestion Topology



6.2.1.4 Ingestion Mechanism

As previously highlighted in the conceptual view chapter there are a number of data movement and data processing mechanisms. Together with the appropriate ingestion topology, these ingestion mechanisms can be utilized for ingestion purposes. Figure 6–7 below highlights how these mechanisms can address one or more data latency requirements.

Figure 6–7 Ingestion Mechanisms



The following table assists an architect in deciding which mechanism(s) should be utilized in addressing their ingestion requirements.

Table 6–5 Ingestion Characteristics

Ingestion Mechanism	Characteristics
ELT	▪ Parallel database centric batch and micro-batch oriented ingestion ▪ Large structured data sets where source and target stores are well defined and modeled ▪ Data quality (e.g. standardization) and complex transformations are well supported
ETL	▪ Batch and micro-batch oriented ingestion ▪ More compute intensive than ELT ▪ Process / Transformation row by row from various data stores.

Table 6–5 (Cont.) Ingestion Characteristics

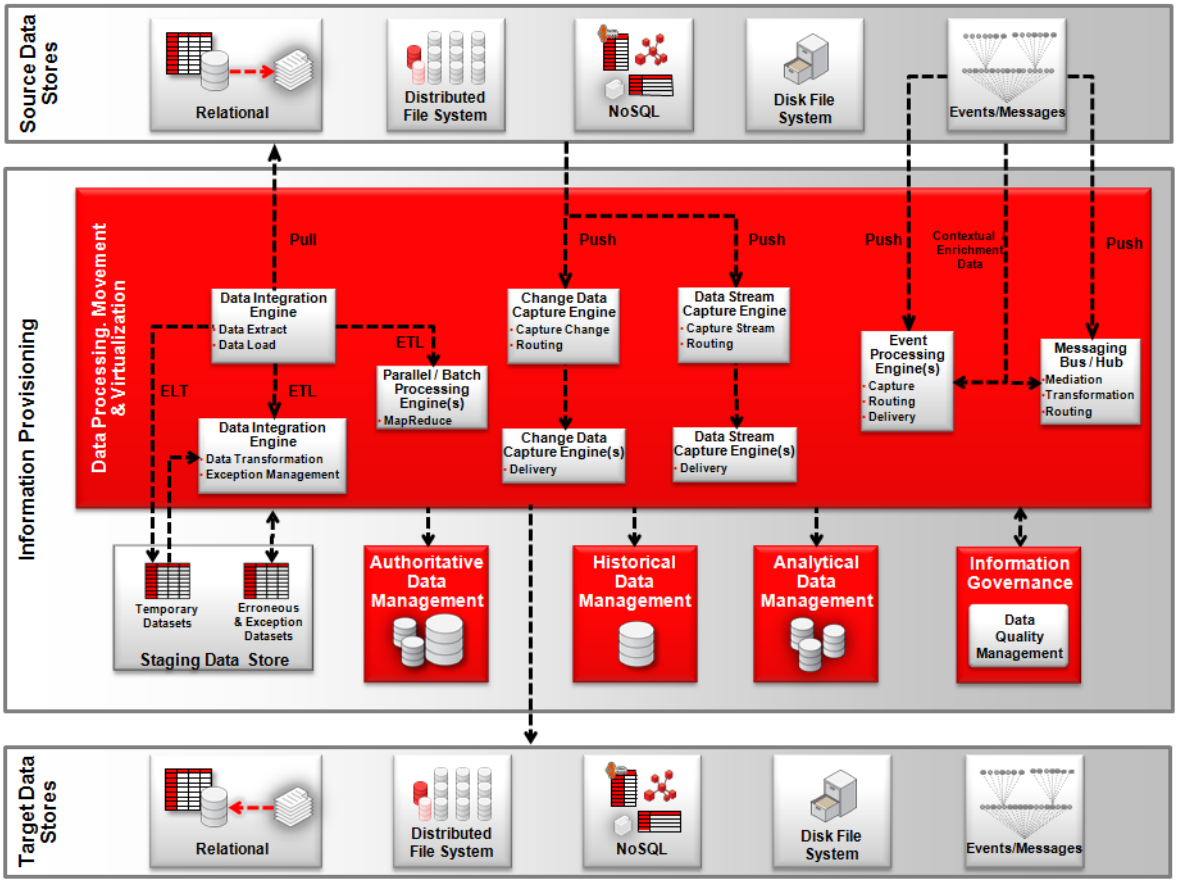
Ingestion Mechanism	Characteristics
ETL (MapReduce)	■ Parallel batch and micro-batch oriented ingestion ■ Ingests extremely large raw data sets with flexible schema ■ Utilizes distributed file systems and NoSQL stores ■ Little or no data quality performed
Data Stream Capture	■ Continuous trickle oriented ingestion ■ Incrementally ingest unstrucutred/semi-structured data streams as generated (e.g. log files) ■ Minimal processing applied
Change Data Capture	■ Continuous trickle oriented ingestion ■ Incrementally ingest transactional changed records ■ Primarily relational data sources ■ Push minimizes latency
Event Stream Processing	■ Continuous trickle oriented ingestion ■ Process event streams in a low latency manner ■ Can be utilized for filtering, aggregation, and masking ■ Further upstream processing
Messaging	■ Continuous trickle oriented ingestion ■ Process small messages in a low latency manner ■ Mature message exchange patterns

The following is a sample list of architectural requirements that should be considered when deciding which ingestion topology, engines and patterns to utilize.

- Filtering Requirements
- High Availability Requirements
- Performance Requirements
- Transformation Requirements
- Security Requirements
- Data Quality Requirements
- Read-only Requirements
- Data Volumes and Retention Requirements (current and projected)
- Network Latency
- Data Ingestion Latency
- Data Variety
- Regulatory Requirements (e.g. inestion is not permitted)

Figure 6–8 below highlights the interactions between key components to support these various data ingestion mechanisms.

Figure 6–8 Ingestion - High level logical diagram



The Information Provisioning Layer in the diagram highlights the use of a number of key engines to support continuous, incremental and bulk data ingestion techniques. In addition, interaction with the data quality management capability is highlighted to support any data quality requirements as part of a data ingestion process.

6.2.1.4.1 Data Integration Engine

Provides ETL/ELT ingestion mechanisms that traditionally have supported the ingestion of bulk structured data from various sources. This engine also provides the flexibility of having a blended approach also known as ETLT where the appropriate mechanism is utilized depending on requirements such as data ingestion volumes, performance, bandwidth, and access to skill-set. ELT/ETL is known as pull techniques which pull data from the source data stores in a less intrusive manner. Example source data sources include databases, message queues and flat files.

The ELT approach has performance advantages by utilizing a database centric architecture that provides a flexible and optimized approach to data ingestion. An ELT approach places the extracted data into a staging data store where the relevant transformations take place. Then data are ingested into the target data store. Rather than process transformation in sets like ELT does, ETL processes transformation on a record by record basis.

The data integration engine can ensure the consistency of the data is controlled by the use of high-level constraints and rules. For a more comprehensive approach to increasing the quality of data, data quality processing can be introduced as part of the ingestion.

For performance reasons, it is sometimes prudent to unload data from a DB to a flat file and then use a DB loader (as denoted by the red arrow). This can be a more efficient method when dealing with large volumes across heterogeneous technologies

6.2.1.4.2 Parallel Batch Processing Engine

A number of parallel batch processing engines currently exist. The current popular choice is a programming model called MapReduce. MapReduce is a share-nothing data processing architecture and parallel programming model. Utilizing MapReduce as part of an ingestion architecture is sometimes known as extreme ELT. MapReduce supports ingestion, filtering, summarization, and transformation of extremely large unstructured /semi-structured data sets utilizing distributed nodes while hiding the complexity of distribution and fault tolerance. A common use case focuses on transforming data before loading the data into a Data Warehouse.

6.2.1.4.3 Change Data Capture Engine

Provides the ability to ingest only the changed records from a source data store, typically transactional inserts, updates, and deletes, reducing the time and resources needed for the ingestion of data. The Change Data Capture mechanism is known as a push technique whereby the source data store pushes the data when a change occurs, which tends to minimize latency. There are several approaches to enable this "push", e.g. log mining, database triggers, and adapters. The changed data are then routed, potentially transformed, and delivered to the target data store.

6.2.1.4.4 Data Stream Capture Engine

While the change data capture engines focus primarily on relational data stores, the Data Stream Capture Engine enables the ability to ingest unstructured/semi-structured data as it is generated in a low impact, low latency manner (e.g. ingestion of log files). The engines consume data from various sources, optionally perform some light-weight processing, and distribute the data either to a target store (e.g. NoSQL, distributed file system) or forward the data to another data stream capture engine.

6.2.1.4.5 Event Processing Engine

Enables the ability to ingest events as they are generated in a low impact, low latency manner. The engine can identify and raise significant events found in the event streams, it does this by performing capabilities such as filtering useless and redundant data (e.g. data ticks with no change), aggregation and basic pattern matching on single streams. This early filtering reduces network load and improves scalability and responsiveness to downstream systems. In addition, the masking of specific elements of an event assists with complying with governance policies such as privacy regulations. More complex capabilities such as correlations across multiple events streams are usually performed by Event Processing Engines downstream. (See Section "Big Data Processing").

6.2.1.4.6 Messaging Bus/Hub

Enterprise Application Integration (EAI) is an approach for integrating multiple applications. EAI products are built around messaging products and are deployed in either a hub-and-spoke architecture or in a bus architecture.

While service-oriented integration utilizes a bus architecture based on standardized communication protocols such as Web services or REST.

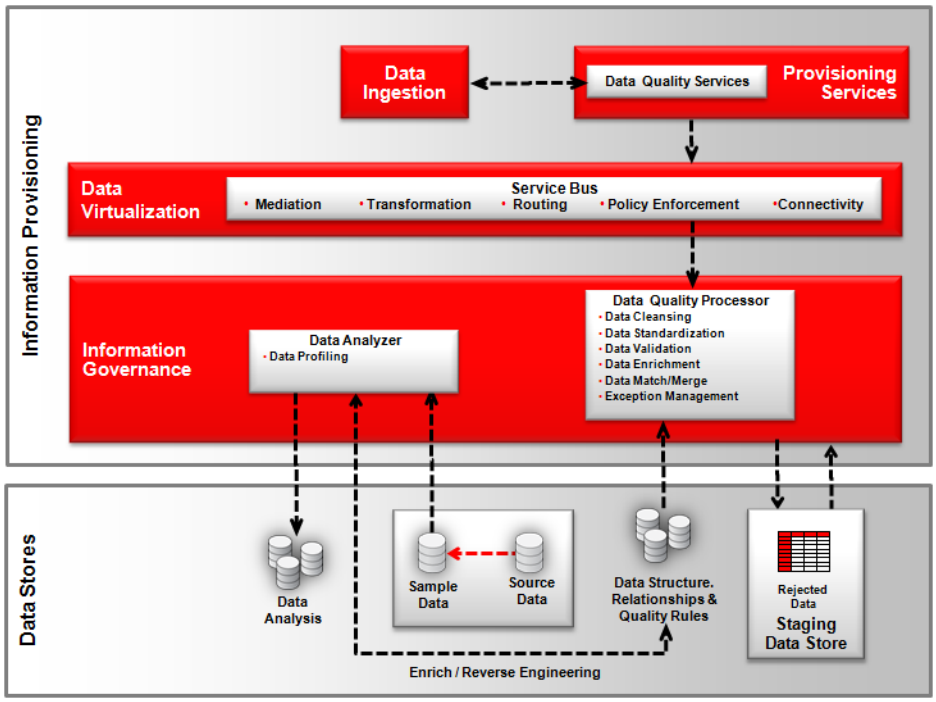
Refer to *ORA Integration* documenr for more details

6.2.1.5 Data Quality Management

As mentioned earlier in this section, Data Quality Management can be utilized to increase the quality of data during a data ingestion process. Obviously it is best to address data quality issues at the source, but if defects can't be prevented at the source it is best to address these issues within a staging area before it spreads to the target data stores.

Figure 6–9 below highlights the interactions between key components to support data quality management.

Figure 6–9 Data Quality Management - High level logica diagram



Data quality management enables the discovery, monitoring, repair, and management of enterprise data stored in relational databases and data files. The Information Provisioning Layer in the diagram highlights the two key components of applying data quality processing.

6.2.1.5.1 Data Analyzer

A Data Analyzer assesses and profiles common data defects on which a default set of rules can be defined. For performance reasons, this analysis can be applied against a sample snapshot of the data rather than on the production data store (as denoted with the red arrow). This analysis allows business users and data architects to assess the quality of their data through data analysis metrics. This enables the data stewards the ability to define new, and enrich existing, data quality rules that a data quality processor can utilize to repair the defects.

6.2.1.5.2 Data Quality Processor

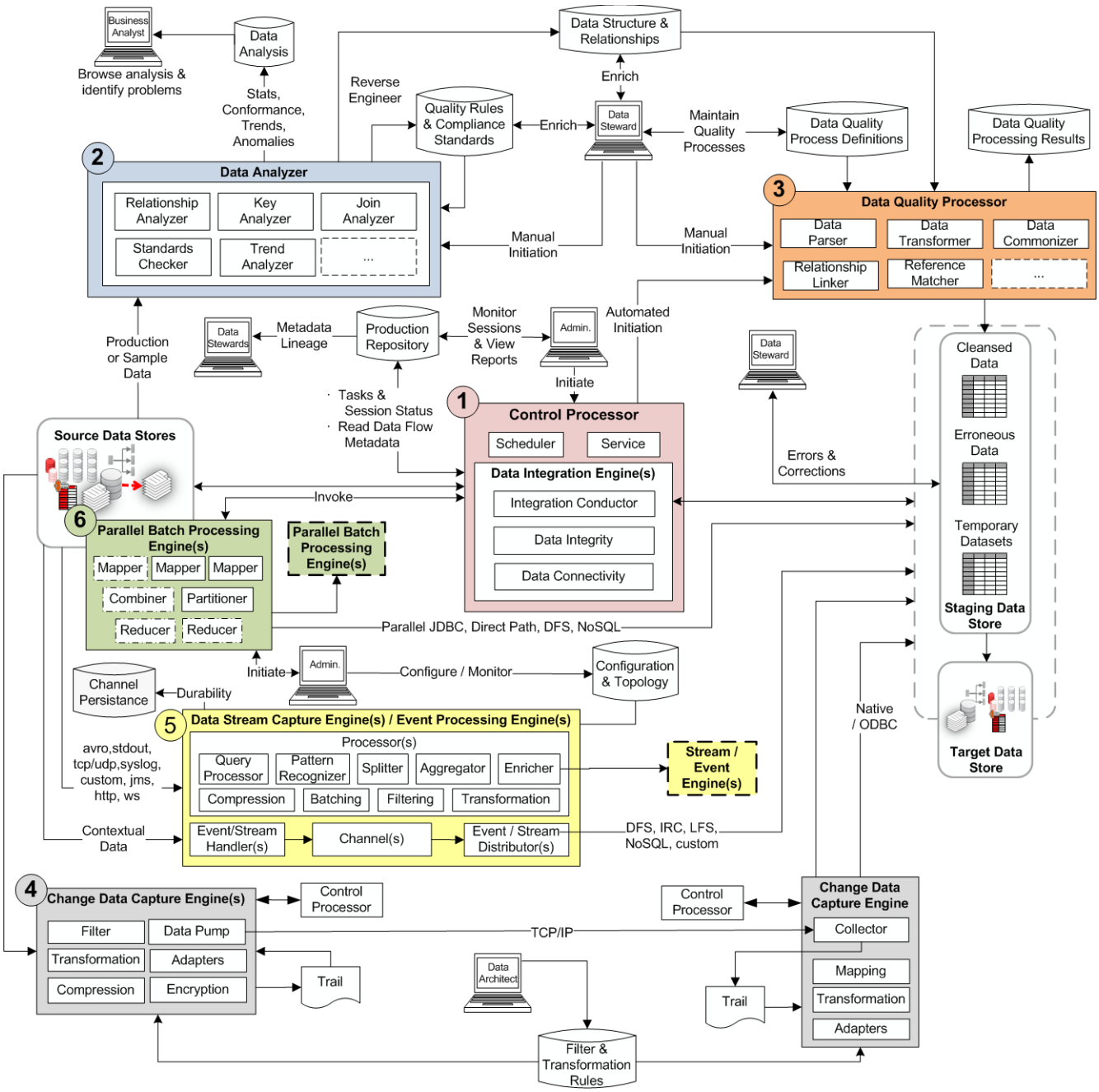
A Data Quality Processor is a rules-based engine that utilizes the data quality rules to repair the data either statically (data at rest) or as part of a data ingestion flow (data in flight). Any defects encountered that cannot be rectified are placed into the staging area where it can be assessed, addressed, and recycled. Initiating the data quality

processor can be manual or via a service. Initiation via a standard service interface is favored over a utilizing a direct API (to support loose coupling).

6.2.1.6 Interactions

More detailed interactions between the key engines and data quality management can be seen in Figure 6–10 below.

Figure 6–10 Ingestion - Interactions



As noted earlier in this section there are a number of mechanisms to support data ingestion. These mechanisms are addressed by six components shown in Figure 6–10 which have their own characteristics and appropriate usage.

1. Data Integration Engine - An ETL/E-LT process can be initiated manually via the console or a command line, on a schedule time basis, or via a information provisioning service (See Section "Information Provisioning as a Service" for more details regarding information provisioning services).

   When an ETL/E-LT process is initiated, a data integration engine is instantiated by a control processor. The data integration engine retrieves the associated execution metadata from the production repository, connects to the source and target data servers, and orchestrates the code execution on these nodes.

   The integration conductor is in charge of orchestrating loading source data to the staging area. Data are extracted from the source data store (e.g., databases, files, message queues) via several possible techniques (e.g. flat file loaders, pumps/links, SQL). Usually a strategy that leverages the most efficient loading utility available for the staging area technology is taken; i.e. most RDBMSs have fast loading utilities to load flat files into tables.

   The integration conductor then orchestrates the loading of the data into the staging area. In the staging area any necessary transformation can take place utilizing the inherent database infrastructure capabilities such as database procedures and SQL commands.

   The data are checked for consistency during extraction and/or loading. Any data that fails consistency checks are placed into an error table. Consistency checks usually include constraints such as primary/foreign/alternate keys, check conditions, and mandatory columns. In some cases, it is useful to recycle errors from previous runs so that they are added to the flow and applied again to the target. This method can be useful for example when receiving daily sales transactions that reference product IDs that may not exist.

   Suppose that a sales record is rejected in the error table because the referenced product ID does not exist in the product table. This happens during the first run of the interface. In the meantime the missing product ID is created by the data steward. Therefore the rejected record becomes valid and should be re-applied to the target during the next execution of the interface.

   Data are then moved into the target data store, (e.g. databases, files, message queues). There are obvious performance advantages when the staging and target data stores are on the same server. For example, a direct INSERT/SELECT (sometimes known as intra-ETL) statement leveraging and containing all the transformations required can copy the data from the staging area to the target.

2. **Data Analyzer** - While it is common for data integration engines to have a level of data analysis capabilities, a dedicated data analyzer provides a deeper, more complete analysis of both the data and its structural characteristics. This assists in identifying mismatches and inconsistencies between the metadata and the actual data. In addition, a data analyzer provides the ability to verify the quality of the data against corporate standards and compliance requirements.

   The data analyzer automatically reverse engineers and analyzes the data and its structure to discover joins, keys, relationships, and dependencies, to assist with identifying data anomalies, broken data quality rules, misaligned data relationships, business rules, and other concerns. An initial set of data quality rules can be discovered and deduced based on the results of this data analysis. Once the data problems are well understood by the data stewards, they can refine, extend, and define new data quality rules to repair the data quality problems encountered. After the data quality rules have been fine-tuned and tested against data samples, the data quality rules can be utilized by data quality processors to repair the defects.

As previously stated, the Data Analyzer can initially work on a sample snapshot of the data rather than on the production data store. But a data steward can connect directly to a production data store so that data analysis can take place in real time. This analysis should not be seen as a one-off effort but be performed over a period of time. The Data Analyzer can continually monitor data conditions and compare snapshots, enabling business analysts to see trends in data usage and identify anomalies as well as areas for improvement.

3. **Data Quality Processor** - A Data Quality Processor is a powerful rules-based engine that executes a number of processes that cleanse, standardize, enrich, match, and de-duplicate any type of data, including names and addresses. Example processes include:

   - **Data Parser** - Uses pattern-recognition to identify, verify, and standardize components of free-form text.
   - **Commonizer** - Defines a golden record from a matched set of records, taking into consideration any survivorship rules.
   - **Reference Matcher** - Augments and corrects data using external data sources. For example - enriching address data by matching data to postal directories and then populating the appropriate attributes with additional or corrected values.
   - **Transformer** - Scans data for defined masks and literal values, and then moves, recodes, or deletes the data.
   - **Relationship Linker** - Recognizes relationships within data and identifies whether duplicate records exist.

   When initiated by a Data Integration Engine, a flat file is generated by the Data Integration Engine which contains the data to clean. This file is passed to the Data Quality Processor which cleanses the data and returns a flat file with the cleansed data.

4. **Change Data Capture Engine** - - Incremental data ingestion primarily is achieved through the use of a Change Data Capture Engine. Change Data Capture is accomplished by capturing just the changed records and not the full data set and delivering this to the target data store, thus dramatically reducing time and resources for the ingestion of data.

   To keep the target source up to date the change data capture engine captures changes made to the source systems, typically transactional inserts, updates, and deletes. Data architects configure the specific data objects that should be captured and synchronized and which technique(s) to utilize to capture the changes (e.g., triggers, log mining, adaptors).

   The engine stores the changes until it receives a commit or rollback. When a rollback is received, the engine discards the data for that transaction. When a commit is received, the engine sends the data for that transaction to a change file (aka trail file), which in turn is routed to the relevant target(s)

   Writing the changed data to a file rather than simply routing the change directly to the target allows for target and network failures. When connectivity to the target is restored, the changed data can be routed to the relevant target.

   Routing is achieved via a data pump which reads this trail file and sends the data over the network, (optionally compressed and encrypted), to a remote trail file on the target. The data pump can perform data filtering, mapping, and conversion.

A target source may require synchronizing multiple source databases to a central target database. Each source system will have a change data capture engine and each will make use of a data pump to send the data to a trail on the target system.

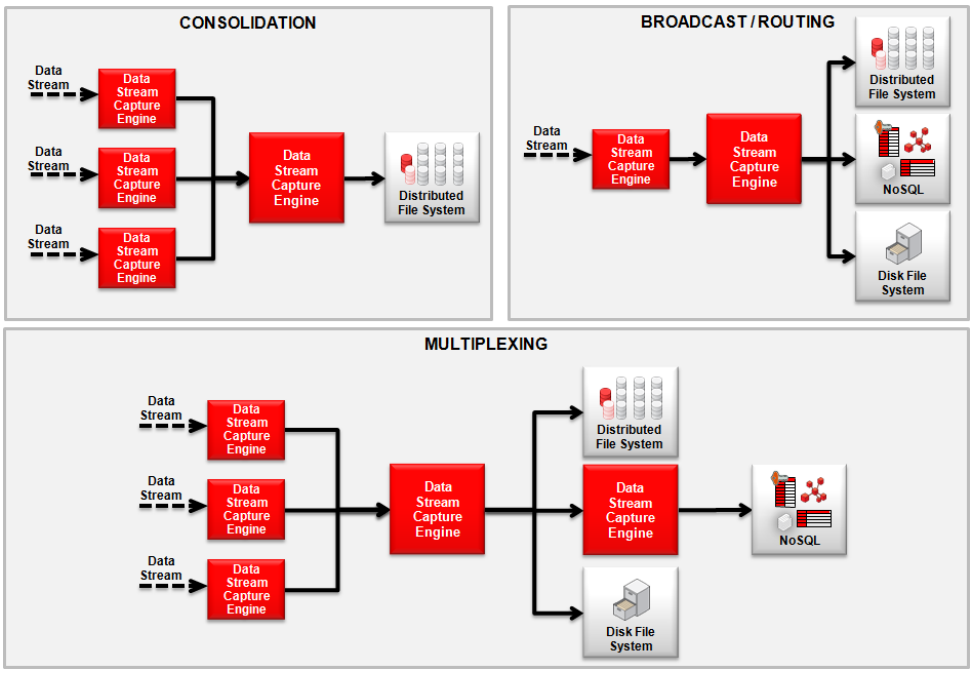
A change data capture engine acting as a collector on the target system receives the extracted database changes that are sent across the TCP/IP network, and writes them to a trail file. The engine reads the trail file and applies the changes to the target system using a native database interface or ODBC, depending on the database type. To preserve data integrity, the engine applies the replicated changes in the same order as they were committed to the source databases. For initial data loads, the change data delivery engine can apply the changes to the data objects in the target database or route them to a high-speed bulk-load utility. Any errors that are encountered when applying the change data to the target data store are captured and written to an error table. The error table includes the original changed data and additional information about the error to assist with resolving the exception.

5. **Data Stream Capture Engine & Event Processing Engine** - The Data Stream Capture Engine and Event Processing Engine offer similar capabilities when utilized as an ingestion engine. Both engines ingest high volume low latency streams. The Data Stream Capture Engine is a distributed engine that efficiently captures, aggregates, and ingests large amounts of streamed data such as log data. For example, rather than batch loading log data on a daily basis, a Data Stream Capture Engine provides a method to ingest data into a target store, such as a distributed file system, as it is generated. The Event Processing Engine captures, aggregates, and ingests large amounts of streamed events, such as those generated by sensors. More details of the Event Processing Engine can be found in Section "Big Data Processing".

   The Data Stream Capture Engine ingests data from various sources, optionally performs some transformation, and distributes the data either to a target store (e.g. distributed file system), or forwards the data to another Data Stream Capture Engine.

   This ability to forward to additional Data Stream Capture Engines enables the definition of flexible ingestion topologies to address requirements such as scalability, routing, consolidation, reliability, etc. See Figure 6–11 for some example patterns.

Figure 6–11 Example Data Stream Topologies



A common use case is to define a data stream topology that ingests log files from various systems into a distributed file system (utilizing the consolidation pattern). But this topology could be extended to also ingest data (full or selective) to another data store (known as a broadcast/fan-out pattern), where the data will be processed/analyzed in a different fashion.

The engine consumes data from various sources (e.g. tail of a text file), optionally performs some processing, and then asynchronously stores the data into one or more channels. The channel is a store that keeps the stream data until it is asynchronously consumed by a sink. The channel can enable the durability of stream data so that if a failure was to occur then the stream data has been stored on disk. The channel also has the option to utilize memory; while this will be faster it will not give any level of recoverability.

The sink removes the event from the channel and stores the data on a target store (e.g. distributed file system), or forwards the data to another stream capture engine.

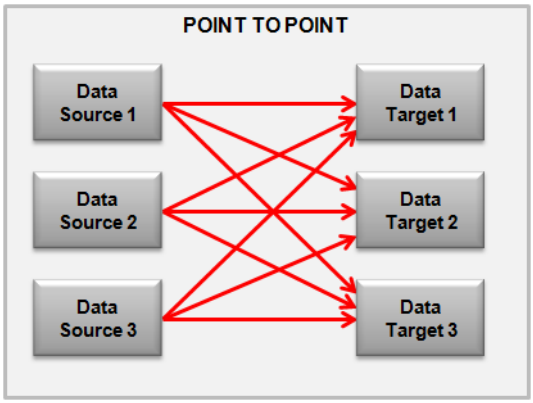
6. **Parallel/Batch Processing Engine** - Parallel/Batch Processing engines can address the ingestion challenges arising from the scale of the data (e.g. volume and velocity), the varied formats (unstructured and semi-structured), and the complex techniques needed to ingest it. For example, each node can process data that is stored on that node. This tends to alleviate the necessity to move large data sets over the network. As previously mentioned, MapReduce is a popular parallel/batch processing engine. More details of MapReduce can be found in Section "Big Data Processing".

6.2.1.7 Ingestion Patterns

The previous sections have highlighted some capabilities, topologies and engines that can be utilized as part of an overall ingestion strategy. This section now details a number of approaches that can be employed to support an overall ingestion strategy.

6.2.1.7.1 Point to Point It is not uncommon for organizations to take the simplest approach to implementing ingestion and employ a point to point approach. See Figure 6–12 below.

Figure 6–12 Point to Point Ingestion



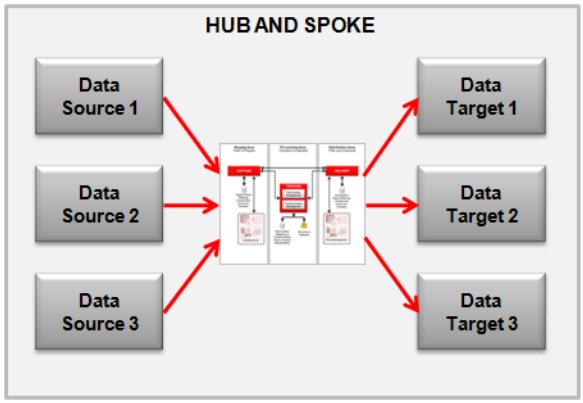
Point to point ingestion employs a direct connection between a data source and a data target. Point to point ingestion tends to offer long term pain with short term savings. That is not to say that point to point ingestion should never be used (e.g. short term solution or extremely high performance requirements), but it must be approved and justified as part of an overall architecture governance activity so that other possibilities may be considered. Otherwise point to point ingestion will become the norm.

Point to point data ingestion is often fast and efficient to implement, but this leads to the connections between the source and target data stores being tightly coupled. In the short term this is not an issue, but over the long term, as more and more data stores are ingested, the environment becomes overly complex and inflexible. For instance, if an organization is migrating to a replacement system, all data ingestion connections will have to be re-written.

Overall, point to point ingestion tends to lead to higher maintenance costs and slower data ingestion implementations.

6.2.1.7.2 Hub and Spoke A common approach to address the challenges of point to point ingestion is hub and spoke ingestion. See Figure 6–13 below.

Figure 6–13 Hub and Spoke Ingestion



The hub and spoke ingestion approach decouples the source and target systems. The ingestion connections made in a hub and spoke approach are simpler than in a point to point approach as the ingestions are only to and from the hub. The hub manages the connections and performs the data transformations.

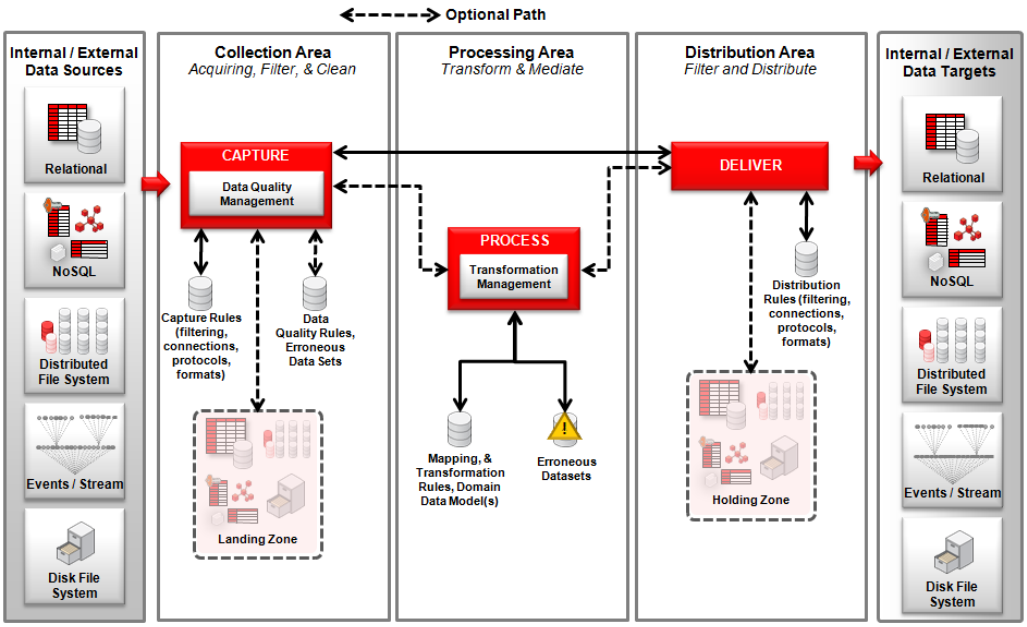
The hub and spoke ingestion approach does cost more in the short term as it does incur some up-front costs (e.g. deployment of the hub). But, by minimizing the number of data ingestion connections required, it simplifies the environment and achieves a greater level of flexibility to support changing requirements, such as the addition or replacement of data stores.

Another advantage of this approach is the enablement of achieving a level of information governance and standardization over the data ingestion environment, which is impractical in a point to point ingestion environment.

It must be remembered that the hub in question here is a logical hub, otherwise in very large organizations the hub and spoke approach may lead to performance/latency challenges. Therefore a distributed and/or federated approach should be considered. To assist with scalability, distributed hubs address different ingestion mechanisms (e.g. ETL hub, event processing hub). Whereas, employing a federation of hub and spoke architectures enables better routing and load balancing capabilities. Invariably, large organizations' data ingestion architectures will veer towards a hybrid approach where a distributed/federated hub and spoke architecture is complemented with a minimal set of approved and justified point to point connections.

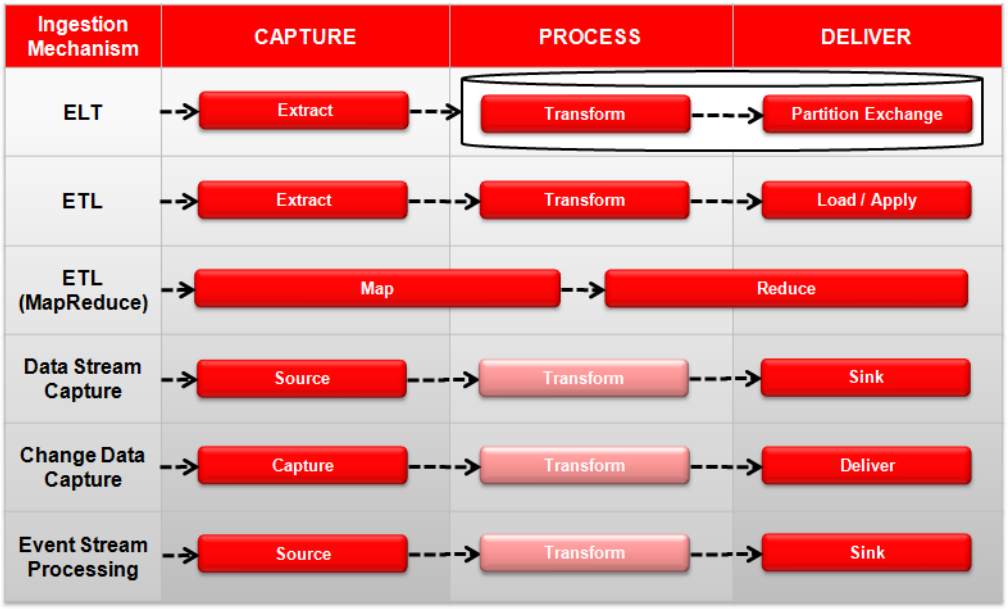
Figure 6–14 below highlights the areas, zones, and processes of an example hub and spoke ingestion architecture.

Figure 6–14 High level hub and spoke architecture



See the figure below for an illustration on how each of the previous detailed ingestion mechanism can be applied to the above ingestion architecture.

Figure 6–15 Ingestion mechanisms applied to hub and spoke architecture



- **Collection Area** - The collection area focuses on connecting to the various data sources to acquire and filter the required data. This is the first destination for acquired data that provides a level of isolation between the source and target systems.

  This capture process connects and acquires data from various sources using any or all of the available ingestion engines. Data can be captured through a variety of synchronous and asynchronous mechanisms. The mechanisms taken will vary depending on the data source capability, capacity, regulatory compliance, and access requirements. The rate and frequency at which data are acquired and the rate and frequency at which data are refreshed in the hub are driven by business needs.

  If multiple targets require data from a data source, then the cumulative data requirements are acquired from the data source at the same time. This minimizes the number of capture processes that need to be executed for a data source and therefore minimizes the impact on the source systems. Looking at the ingestion project pipeline, it is prudent to consider capturing all potentially relevant data.

  If a target requires aggregated data from multiple data sources, and the rate and frequency at which data can be captured is different for each source, then a landing zone can be utilized. The landing zone enables data to be acquired at various rates, (e.g. in small frequent increments or large bulk transfers), asynchronous to the rate at which data are refreshed for consumption. The data captured in the landing zone will typically be stored and formatted the same as the source data system. The stores in the landing zones can be prefixed with the name of the source system, which assists in keeping data logically segregated and supports data lineage requirements.

  If required, data quality capabilities can be applied against the acquired data. Performing this activity in the collection area facilitates minimizing the need to cleanse the same data multiple times for different targets.

- **Processing Area** - The processing area enables the transformation and mediation of data to support target system data format requirements. This requires the

processing area to support capabilities such as transformation of structure, encoding and terminology, aggregation, splitting, and enrichment. In addition, the processing area minimizes the impact of change (e.g. change of target and/or source systems data requirements) on the ingestion process.

If both the source and target systems use the same format for the data, and no transformation is required, then it is possible to bypass the processing area. This is quite common when ingesting un/semi-structured data (e.g. log files) where downstream data processing will address transformation requirements.

As previously stated, the intent of a hub and spoke approach is to decouple the source systems from the target systems. This means not only decoupling the connectivity, acquisition, and distribution of data, but also the transformation process. Without decoupling data transformation, organizations will end up with point to point transformations which will eventually lead to maintenance challenges.

To circumvent point to point data transformations, the source data can be mapped into a standardized format where the required data transformations take place, upon which the transformed data is then mapped onto the target data structure. This approach does add performance overhead but it has the benefit of controlling costs, and enabling agility. This is achieved by maintaining only one mapping per source and target, and reusing transformation rules.

This standardized format is sometimes known as a canonical data model. It is independent of any structures utilized by any of the source and target systems. It is advantageous to have the canonical data model based on an enterprise data model, although this is not always possible. The enterprise data model typically only covers business-relevant entities and invariably will not cover all entities that are found in all source and target systems. Furthermore, an enterprise data model might not exist. To address these challenges, canonical data models can be based on industry models (when available). This base model can then be customized to the organizations needs.

While it is advantageous to have a single canonical data model, this is not always possible (e.g. cost, size of an organization, diversification of business units). In this instance a pragmatic approach is to adopt a federated approach to canonical data models. For example, each functional domain within a large enterprise could create a domain level canonical data model. Transformations between the domains could then be defined.

- **Distribution Area** - The distribution area focuses on connecting to the various data targets to deliver the appropriate data.

  This deliver process connects and distributes data to various data targets using a number of mechanisms. Data can be distributed through a variety of synchronous and asynchronous mechanisms. The mechanisms utilized, and the rate and frequency at which data are delivered, will vary depending on the data target capability, capacity, and access requirements.

  Initially the deliver process acquires data from the other areas (i.e. collection, processing). This data can be optionally placed in a holding zone before distribution (in case a "store and forward" approach needs to be utilized).

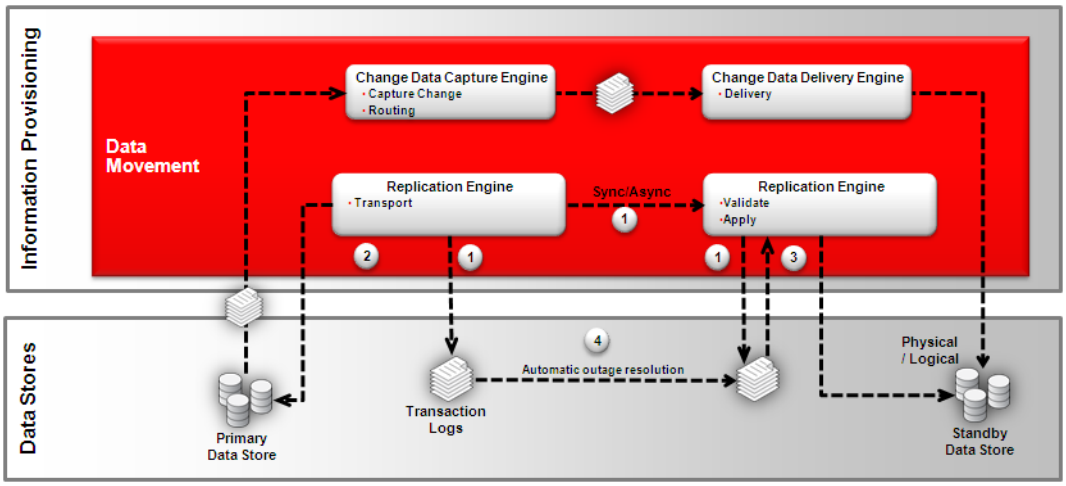
  The deliver process identifies the target stores based on distribution rules and/or content based routing. This can be as simple as distributing the data to a single target store, or routing specific records to various target stores.

6.2.2 Replication & Synchronization

Replication & Synchronizationenables the creation and maintenance of one or more synchronized standby data stores that protect data from failures, disasters, errors, and corruptions. It can address both high availability (HA) and disaster recovery (DR) requirements. In addition, it can address off-loading reporting requirements from the production data stores.

Figure 6–16 below highlights the interactions between key components to support replication and synchronization.

Figure 6–16 Replication & Synchronization - High Level Logical



6.2.2.1 Change Data Capture Engine

It is possible to address replication and synchronization by creating a standby data store from a backup of the primary data store and then utilizing an incremental data ingestion engine such as a change data capture engine. This is best suited when the source and target sources are heterogeneous and information distribution is the primary requirement.

6.2.2.2 Replication Engine

When the database being sourced and targeted are homogeneous and requirements are focused on high availability and disaster recovery, then the use of a replication engine is more suited. A standby database is created from a backup of the primary database, and then the replication engine keeps the data synchronized.

1. As transactions are committed to the primary data store the transaction log records are transported (synchronously or asynchronously) to a remote replication/synchronization engine. The received transaction log is then written to a standby transaction log file. An acknowledgement is then sent to confirm that the transaction log was received and captured to disk.

   There are two types of transport: asynchronous and synchronous.

   - **Asynchronous** - avoids impact to primary data store performance as the primary data store does not wait for acknowledgement from the standby data store that it has successful received the transaction log records. The performance benefit of an asynchronous approach, however, is accompanied by the potential for a small amount of data loss since there is no guarantee that all transaction log records have been received by the standby data store. For a

zero data loss solution, the transaction logs should be transported synchronously.

- **Synchronous** - Synchronous transport causes the primary data store to wait for confirmation from the standby data store that the transaction log file has been written to disk before it will acknowledge transmission success. Without the proper architecture the synchronous transport approach can have a noticeable impact on performance due to the time it takes for the network round trip and writing of the transaction log file. To reduce this impact the architecture must support the transportation of the transaction log file, and the primary data store writing to a local file, being performed in parallel.

  Table 6–6 will assist an architect to determine the best transport approach.

Table 6–6 Transport & Replication Requirements

Requirements	Transport	Acknowledgement	Risk
Maximum Protection	Synchronous	Stall primary data store until acknowledgement from the standby data store is received	Zero data loss
Maximum Availability	Synchronous	Stall primary data store until acknowledgement from the standby data store is received or a timeout threshold period expires – then processing resumes.	Zero data loss
Maximum Performance	Asynchronous	Primary data store never waits for standby acknowledgment.	Potential for minimal data loss

2. The data are committed to the primary data store.
3. The "Apply" capability reads the standby transaction log file, validates it, and applies it to the standby data store using either a physical or logical apply technique.
   - **Physical** - A physical standby is identical to the primary data store on a block-for-block basis, and thus, the schemas, including indexes, are the same. The Physical Apply is the simplest, fastest, most reliable method of maintaining a synchronized replica(s) of a primary data store.
   - **Logical** - A logical standby data store contains the same logical information as the primary data store, although the physical organization and structure of the data can be different. Logical Apply keeps a logical standby synchronized by transforming the transaction log records received from the primary data store into SQL statements and then executing the SQL statements on the standby data store.
4. In cases where the primary and standby data stores become disconnected, (via network failures or standby server failures), the primary data store will continue to process transactions and accumulate a backlog of transaction log files that cannot be shipped to the standby until a new connection can be established. When a connection is re-established, the standby data store will then automatically resynchronize with the primary.

The transport and apply services are completely independent. The status or performance of the standby apply has no impact on the transaction log transport or primary data store performance. This isolation is very important. Transport in

synchronous configurations is the chief determinate of impact to primary data store response time and throughput. Anything that negatively impacts transport in a synchronous configuration can reduce the primary data store throughput and increase response time.

Traditionally organizations have resorted to mirroring techniques to provide the data replication and synchronization capabilities. The issue with such an approach is that a mirroring approach continually replicates every write made to every file, and does so in write-order, in order to maintain real-time synchronization of a remote replica. But a more efficient approach is to only replicate the writes on the transaction log file as tests have shown that mirroring transmits up to 7 times the volume.

6.2.2.3 Replication Engine or Change Data Capture Engine?

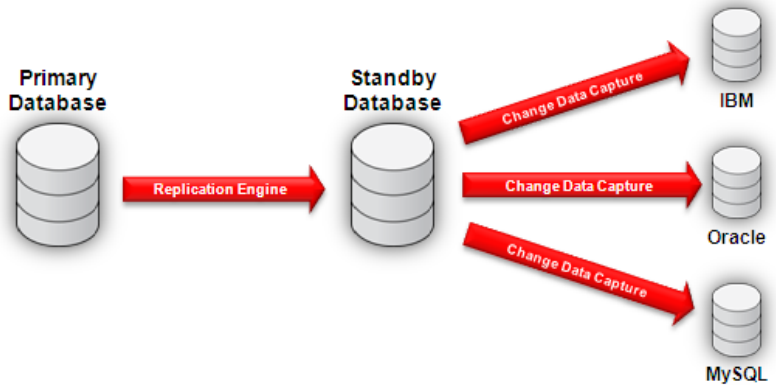
It is up to the architect to decide which replication technique meets their requirements. Each technique has its own appropriate usage and characteristics, see Table 6–7.

Table 6–7 Replication Characteristics

Characteristics	Change Data Capture Engine	Replication Engine
Primary Usage	Information Distribution	Data Protection, HA/DR
Replica	Logical	Physical / Logical
Communication	Asynchronous	Asynchronous/Synchronous
Direction	Bi-Directional	Uni-Directional
Performance	High	Extreme
Target Data Store	Read / Write	Read Only
Target Support	Heterogeneous	Homogeneous
Data Type Support	Restricted by Target Support	Full
Transformation	Yes	No
Corruption Support	Data Consistency Validation	Automatic Block Repair

An architect does not have to choose one of these mechanisms over the other. They are in effect complementary and not mutually exclusive approaches. See Figure 6–17 for an example that depicts the complementary use of a replication engine and a CDC engine.

Figure 6–17 Complementary Replication Mechanisms



In the above example, a replication engine makes available a standby database which can be utilized for data protection and to offload read-only workload from the primary database. The CDC engine is used for heterogeneous replication of different subsets of the standby database to multiple target heterogeneous databases. The CDC engine could have been applied to the production database rather than the standby database but would have added additional overhead to process incremental data movement.

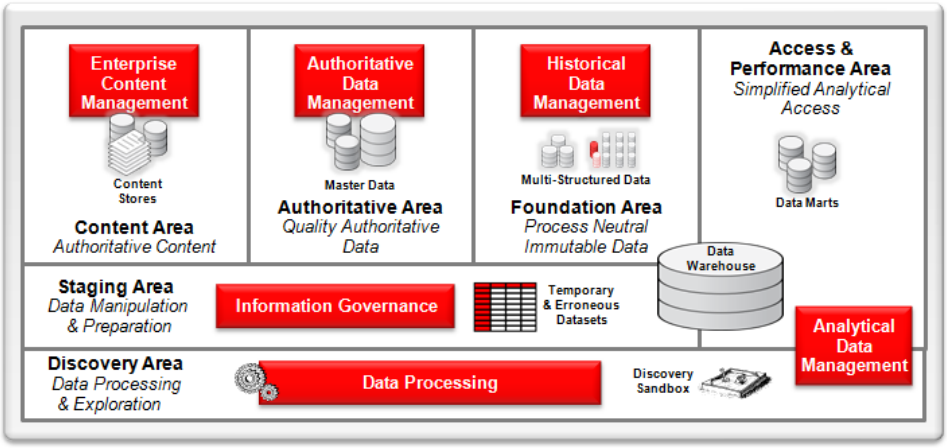
6.3 Information Organization

Information Organization represents a group of scenarios that pertain to provisioning information in various forms, persistence models, and semantics, using industry and enterprise models for various consumers. These scenarios are not tied to a specific solution or consumer; rather, they are designed for broader use and multiple purposes.

6.3.1 Areas of Responsibility

Information Organization as seen in Figure 6–18, has been partitioned into various areas which support and enable various information provisioning capabilities.

Figure 6–18 Information Organization - Areas of Responsibility



These areas support the separation of concerns around logically defined boundaries of responsibilities. These distinct areas not only have distinct capabilities but also have their own logical data stores to support their capabilities and the role they play in an overall information management approach.

6.3.1.1 Staging Area

The Staging Area can act as either a temporary and/or permanent area where data pre-processing occurs for downstream usage, such as manipulation, cleansing, and the production of quality data.

Generally, the Staging Area is the first destination of data that has been acquired. This provides a level of isolation between data that will be moved into the Information Provisioning layer and data that is generally available to consumers.

By providing this isolation, data can be acquired at various rates, (e.g. in small frequent increments or large bulk transfers), asynchronous to the rate at which data are refreshed for consumption. Data can be acquired through a variety of synchronous and asynchronous mechanisms. The mechanisms taken will vary depending on the

data source capability, capacity, regulatory compliance, and access requirements. The rate and frequency at which data are acquired, and the rate and frequency at which data are refreshed in the staging area, is driven by business needs in support of the various information provisioning scenarios (e.g. Master Data Management (MDM), Big Data Processing, and Data Warehousing).

When there are requirements that dictate that data are clean, consistent, and complete (as far as is practical), the Staging Area is where data quality business rules are applied to achieve these objectives.

The Staging Area can contain many data objects such as:

- **Source Tables/Files** - These tables and/or files hold data from various data sources.
- **Decode Tables** - These tables hold the master list to coordinate integration of multiple sources.
- **Lookup Tables** - These tables are reference tables for codes.
- **Erroneous Tables** - These tables hold rejected data that has failed consistency and quality checks. Rejected data are retained in this area for manual or automatic correction.
- **Temporary Tables** - These tables are truncated and reused with every load.

6.3.1.2 Foundation Area

The Foundation Area is responsible for managing data for the long term.

It is in the Foundation Area that data from all originating sources is maintained in a uniform manner. In order to most easily adapt to changes in the organization over time, structured data are maintained here in a business neutral form. Changes to organizational structures and business processes should not impact the way data are stored in this area. Versatility takes precedence over navigation and performance. For unstructured data, the Foundation Area is commonly used as a collection point of historical record.

6.3.1.3 Access and Performance Area

The Access and Performance Area is used to represent data in ways that best serve the analytical community. Since it does not have responsibility for historical data management, it is free to represent data for most efficient access and ease of navigation.

The structure of the Access and Performance Area is dependent upon the tools and applications using the data. For structured data, this will often consist of dimensional models and/or cubes. In addition, it may contain aggregations of facts (rollups) for rapid query responses. Access and Performance Area structures can be instantiated and changed at will in order to conform to the business hierarchies of the day. For unstructured data, this area may be implemented as a subset of historical data with or without additional metadata, indexing, or relationships added that enhance accessibility and query performance.

In addition, the deployment of specialized hardware or software can enable this layer with caching mechanisms and in-memory databases to meet query response time requirements.

6.3.1.4 Authoritative Area

The Authoritative Area is responsible for managing master and key reference entities such as Customer, Supplier, Account, Site, and Product. The data can be managed

within the Authoritative Area either virtually, physically, or a combination of both of these methods. The Authoritative Area maintains master cross-references for every master business object and every attached system.

With all the master data in the Authoritative Area, only one ingestion point is needed into upstream or downstream systems that require the consumption and delivery of these key master and reference entities. This enables the Authoritative Area to support operational data by ensuring quality and providing consistent access for use by operational applications, while also supporting the delivery of consistent historical and analytical reporting.

6.3.1.5 Content Area

The Content Area is responsible for managing and providing access to user generated unstructured/semi-unstructured data.

While data in the Content Area is indexed and searchable, the true value of this data is when it is not just utilized in isolation but in conjunction with other forms of data such as operational data and authoritative data. Therefore the Content Area enables the consumption and access of user generated unstructured/semi-unstructured data to downstream systems.

6.3.1.6 Discovery Area

The Discovery Area provisions and manages a number of investigative data stores (aka Sandboxes) to enable exploratory analysis. The Discovery Area ingests data from various downstream and upstream sources in support of specific business problems. It is also where user-defined data can be introduced for simulation and forecasting purposes.

Within the Discovery Area the data can be used to discover new insights and relationships. Any discovered valuable results are then made available to be ingested into either downstream or upstream systems.

These sandboxes tend to be initiated and owned by data analysts/scientists and typically have a limited lifespan. This data is processed and analyzed using any and all tools at the disposal of the Data Scientist, and therefore can often be resource and processing intensive.

The interaction between these areas varies depending on which information organization scenarios are being addressed. This section will expand on the following information organization scenarios:

- Big Data Processing
- Enterprise Data Management
- Data Warehousing

6.3.2 Big Data Processing

As highlighted in the Concepts chapter, Big Data is a term applied to data sets whose size, structure, and acquisition speed is beyond the capability of conventional software tools used for capturing, ingesting, managing, and processing data.

Combined with the complexity of analysis and commercial imperative to create value from it, Big Data has led to a new generation of technologies, tools, and architectures.

6.3.2.1 Fast Data

Fast Data[1] is a complementary approach that focuses on the velocity side of Big Data. Fast Data addresses the need to acquire and process information in near-real time so that companies can acquire insight into the business in a timely manner and react accordingly. For example, customer experience, fraud detection, smart meters, etc. While Fast Data capabilities are not new, recent trends in technology price/performance have made the capabilities more widelt obtainable.

Within the realm of Big Data, Fast Data can be seen as the acquisition and processing capabilities that address data in motion, where there is a requirement for extremely low latency processing in a high velocity environment.

Fast Data demands special handling, but primarily in terms of system load and performance. Fast Data is often associated with event stream processing and in-memory databases. Therefore the architecture tends to be event-driven - capable of accepting a continuous data stream and taking action on specific patterns. It is also usually memory-resident, since reading and writing to disk is not feasible on a record by record basis.

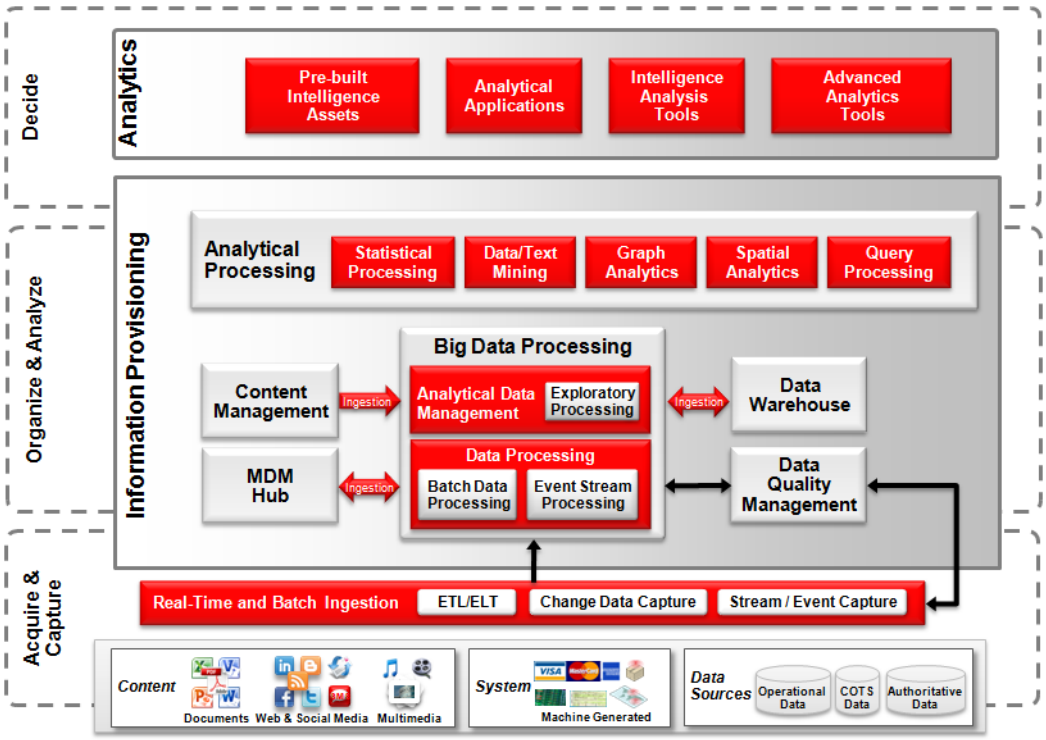
For the remainder of this section the term Big Data will also encompass the capabilities offered by Fast Data.

6.3.2.2 Big Data Processing in Context

The requirements for Big Data Processing focuses on the ingestion, organization and processing of new data sources with existing information stores (e.g. Data Warehouse) so that new and existing analytical infrastructure can take advantage of any newly discovered business value. Figure 6–19 illustrates 3 major clusters of architecture requirements.

[1] "What is Fast Data", Tony Bear, Ovum, Nov 2012

Figure 6–19 Big Data Processing - In Context



- **Acquire & Capture** - Ingesting data for Big Data processing brings different requirements due to the 3V's (Volume, Variety, Variety). The architecture must support the ingestion of Big Data and deliver low latency in both the capturing of data as well as in executing short, simple queries; be able to handle very high transaction volumes, often in a distributed environment; and support flexible, dynamic data structures.
- **Organize** - Traditionally, the majority of data has been stored in relational databases. The architecture requirements to cater for Big Data processing (at-rest and in-motion) highlight the need to support various processing, persistence, and storage models (polyglot persistence) which can then be applied to the appropriate scenarios. Table 6–8 highlights different approaches to Big Data processing for data-at- rest and data-in-motion.

Table 6–8 Data-at-Rest, Data-in-Motion

	Data-at-Rest	Data-in-Motion
Characteristics	Process low density data in batches	Process events at high velocity
Time Criticality	Low - Process and analyze later	High - Process and analyze now
Processing	Distributed on disk processing	In-memory data processing
Engine	Parallel batch processing engine	Event processing engine

For example, to address the "volume" characteristic of Big Data while addressing high throughput processing and manipulation requirements, it is advantageous to organize and process the data into aggregated results while keeping the data in the same storage location.

- **Analyze & Decide** - The architecture must be able to support deeper analytics such as statistical analysis and data mining, on a wider variety of data types stored

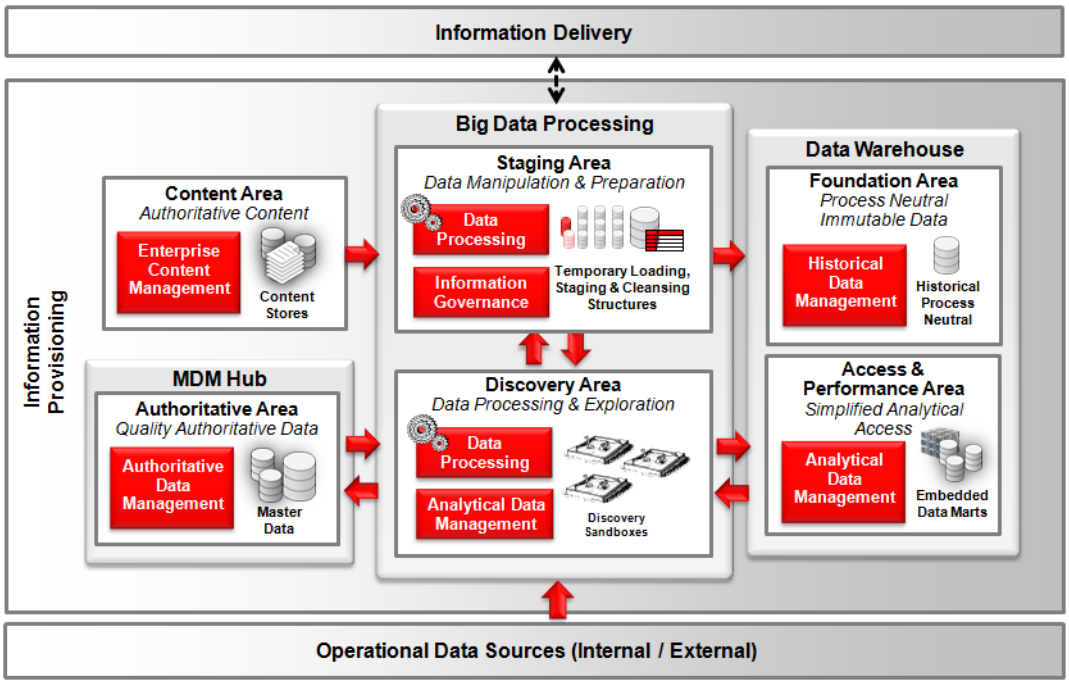
in diverse systems; scale to extreme data volumes; deliver faster response times driven by changes in behavior; and automate decisions based on analytical models.

Figure 6–19 above also highlights the separation of the storage and processing of Big Data (Information Provisioning Layer) from business analytical aspects (Analytics Layer). The remainder of this section focuses on the Big Data Processing aspects of Big Data. For details regarding the analytical aspects refer to the *ORA Business Analytics* documents.

6.3.2.3 Areas of Responsibility

The Information Provisioning Layer, seen in Figure 6–20, has been partitioned into various areas to support and enable Big Data processing. These areas support the separation of concerns around logically defined boundaries of responsibilities.

Figure 6–20 Big Data Processing - Areas of Responsibility



Big Data processing can exploit data found within unstructured and semi-structured data. Therefore, content is a major source for Big Data processing, covering activities such as information discovery and text mining. This content can be ingested from either a controlled Content Area via enterprise content management systems or ingested from uncontrolled or raw internal/external operational data sources.

As well as content being ingested, authoritative data can also be ingested into the Big Data Processing environment. Authoritative data can be utilized to give context to data stream processing to enhance pattern detection and enrichment of events. In addition, authoritative data can be utilized to assist in the cleaning, de-duplication and normalization of data during processing (e.g. integrating customer master data and social media metrics as part of a sentimental analysis process).

Big Data Processing operates in two environments:-

- Data Processing & Exploration
- Data Manipulation & Preparation

Figure X highlights their usage and characteristics of each environment.

Table 6–9 Big Data Processing Environments

Environment	Data Processing & Exploration	Data Manipulation & Preparation
Focus	Discovery sandbox	Formal analytical pre-processing environment
Purpose	Quick discovery and explotation of information in order to justify new business opportunities, or to identify the source(s) of a problem or anomaly.	Pre-processing and insights within an environment that offers consistency, execution optimization, pre-ingested data, and data quality capabilities.
Activities	Ah-hoc activities	Repeatable operationalized activities
Data Sources	Ad-hoc, flexible	Trusted, audited, secured, protected
Data Quality	Primarily raw data but access to cleansed data if required	Appropropriate data quality determined and applied.

6.3.2.3.1 Data Processing & Exploration

A Discovery area is utilized to provision sandbox environments for the purpose of ad-hoc data processing and Exploratory Analytics. Sandboxes can be utilized for the quick discovery and exploitation of information in order to justify new business opportunities, or to identify the source(s) of a problem or anomaly (i.e. Exploratory Analytics).

Exploratory Analytics applies ad-hoc exploratory activities that generally involve the analysis, filtering, distillation, classification, and correlation of data in order to create a useful form of knowledge or understanding.

Organizations commonly utilize an exploratory sandbox to address a business problem as quickly and simply as possible. Following the initial provisioning of a sandbox environment, local ad-hoc data as well as data from identified from internal/external operational source systems are ingested.

When performing data exploration activities, such as business opportunity identification, data relationship research, and data anomaly detection, it is advantageous to have access to raw data (prior to any cleansing taking place). Otherwise there is a risk of cleansing away any analytical value. For example, outliers may represent potential fraud, may infer sensor malfunction, may highlight inconsistent event generation across sensors, and may represent issues in the sensor collection system. Any discovered correlations and outliers assist in forming an initial understanding of the data distribution.

The proper determination of data quality can be addressed once the types of analytics and questions that need to be answered have been identified. Additional data from the Content, Authoritative, Staging, Foundation, and Access and Performance Areas may also be included to add context in support of the business problem being tackled.

If an organization determines that there is some business value to the processing of the underlying data, then the analytical results as well as the processing components can be transitioned from this environment to an environment where it can be formalized and controlled.

Details regarding Exploratory Analytics can be found in the *ITSO Business Analytics* documents.

6.3.2.3.2 Data Manipulation & Preparation

A formal analytical pre-processing environment enables the operationalization of Big Data Processing and insights within an environment that offers consistency, execution optimization, pre-ingested data and additional data quality capabilities. This operationalized environment can be utilized with or without proof of identifiable business value within the new and derived data sources.

Logically, it is advantageous to employ the Big Data Processing capabilities within the Staging and Foundation area, as in many cases, the data required to address any particular business problem will already be present in this environment. If not, the environment can ingest new data which may come from any source or information provisioning area.

Depending on the use of the data (if known), it might make sense to not apply any data quality controls until after Big Data Pre-Processing has completed, when it is being ingested downstream (e.g. to the Data Warehouse).

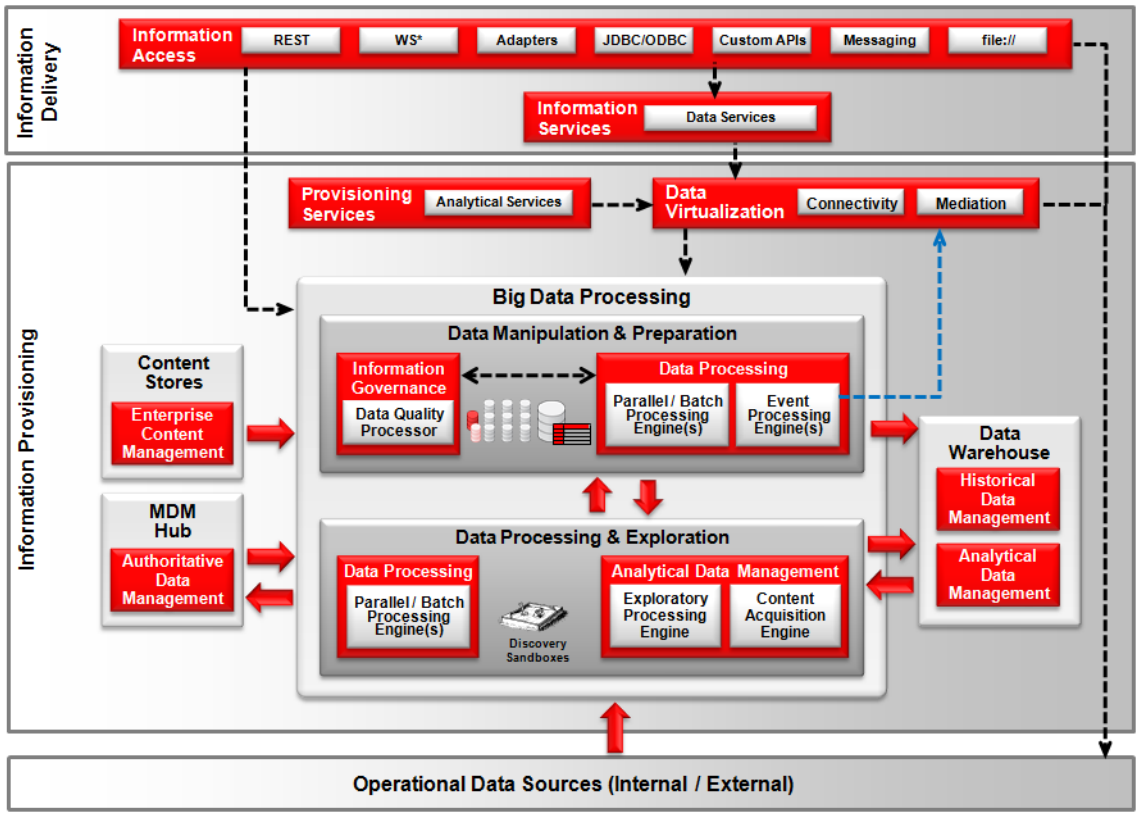
6.3.2.4 Example Interactions

Within a Big Data environment users utilize more complex data visualization tools to glean valuable information from internet activity and business behavior (e.g. clickstream analysis, social media monitoring, and customer sentiment analysis). This requires access to more advanced analytical capabilities than in the past, which in turn requires access to and the processing of extremely large data-sets both in-motion and at-rest.

The architecture must address challenges, such as processing extremely large data sets efficiently, processing streaming data within a moving window, and reliably drawing insights by exploring data that may be part structured, part textual, modeled, and free form.

With the various Big Data Processing requirements, usage patterns, and processing techniques available, there is no single way to architect for Big Data Processing. The rest of this section will focus on an example scenario highlighting data processing mechanisms, components, and interactions points. These are by no means the only interactions, but they highlight the number of options and flexibility that a Big Data Processing solution can offer. Figure 6–21 illustrates the high-level interactions between key components to support Big Data Processing.

Figure 6–21 Big Data Processing - High level logical



Big Data has highlighted the need to include unstructured and semi-structured data as part of an overall data processing environment. This includes both user and machine-generated data. To leverage the full value of this data and to explore business opportunities, companies have to capture this data, not just in isolation on a distributed file system, but pre-processed so that data is indexed, searchable, and can be mined and analyzed in a timely manner.

The Information Provisioning Layer in the above diagram highlights the use of a number of key engines to support ingestion and processing of Big Data. In addition, the diagram illustrates the interactions with a number of key areas such as Data Warehousing and Master Data Management Hubs.

6.3.2.4.1 Data Ingestion

Ingestion for Big Data processing differs from traditional ingestion which has tended to focus on ingesting structured data using ETL mechanisms. Data is being generated with greater velocity and variability than ever before, and, unlike data in the past, most of it tends to be unstructured and raw. If this data can be quickly ingested, processed, and analyzed then companies will be able to uncover new patterns, relationships, and insights that they weren't able to recognize in the past. The red arrows in Figure 6–21 indicate data ingestion. Ingestion of Big Data leverages a number of mechanisms to collect and acquire unstructured and semi-structured data from a variety of data sources into a variety of data stores, where it will later be utilized for downstream Big Data processing. Big Data ingestion tends to focus on acquiring data sets with high velocity, high volume, and higher variety, while delivering low and predictable latency. Ingestion utilizes a number of different engines in the handling of this data. Refer to Section "Ingestion" for more details regarding ingestion.

6.3.2.4.2 Parallel Batch Processing Engine

As highlighted earlier, parallel batch processing engines such as MapReduce can be utilized as part of an ingestion architecture. In addition, MapReduce can be used as a data processing engine. MapReduce can address many data processing needs when the data in question is large and varied enough and where conventional data processing engines may be deemed too slow, expensive, or not flexible enough to obtain the same results. Parallel Batch Processing Engines assist with filtering data through parallel processing and indexing mechanisms, where the results are often correlated and merged with traditional data sources such as Master Data Management and Data Warehousing. For example, the data processing of log data for call detail records analysis, behavioral analysis, social network analysis, and clickstream sessionization. While traditional Data Warehouses have addressed these needs in the past, the growing volume and variety of data has led these data processing needs to be addressed with parallel batch processing engines such as MapReduce. This is not to say that MapReduce is the answer to all data processing needs as MapReduce is primarily a batch processing system. For example, high velocity, time-critical data processing is better suited to an event processing engine

6.3.2.4.3 Event Processing Engine

As highlighted earlier an event processing engine can be utilized as part of an ingestion architecture where the engine tends to focus on performing basic data reduction (filtering), simple aggregation, and distribution on single streams, (where a stream is a real-time, continuous ordered sequence of events). Within a Big Data Processing architecture the Event Processing Engine enables real-time data processing, which supports executing real time decisions by detecting, capturing, and processing actionable events within multiple high volume, time-critical, low latency event streams. The Event Processing Engine continuously queries and processes internal/external event streams performing processing tasks such as pattern matching, filtering, aggregation, correlation across multiple streams, and, if required, raising/distributing identified significant events to another event stream (as shown by the blue arrow). Generally, due to filtering and aggregation, the number of outbound events is much lower than that of the inbound events.

6.3.2.4.4 Exploratory Processing Engine

To support intuitive exploration and analysis of information, an Exploratory Processing Engine provides unprecedented flexibility in combining diverse and changing data as well as extreme performance in analyzing data by utilizing a hybrid, search/analytical database. This exploratory database enables a dynamic faceted data model that evolves depending on the data being ingested and processed. A faceted data model is extremely flexible and reduces the need for up-front data modeling and supports an iterative "model as you go" approach. The Exploratory Processing Engine performs a number of enrichment activities that transform, standardize, and process data from various sources (e.g. operational, pre-crawled content) for the purpose of Exploratory Analytics.

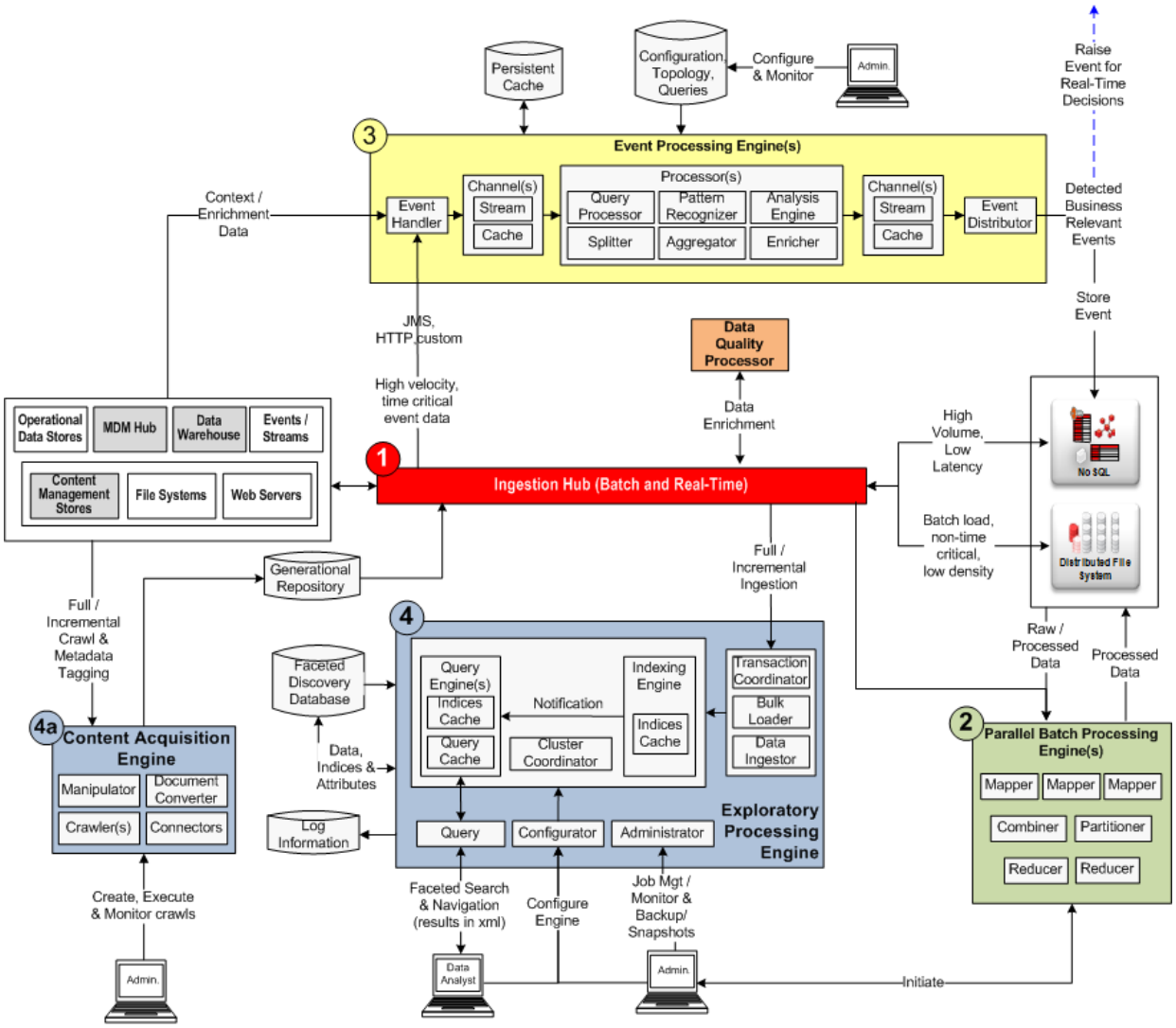
6.3.2.4.5 Content Acquisition Engine

Provisions crawled data from multiple data sources such as file systems, content management systems, and web servers. This crawled data is intelligently tagged and made available for the purposes of exploratory analytics.

6.3.2.4.6 Detailed Interactions

Figure 6–22 below shows a more detailed view of the components and their interactions.

Figure 6–22 Big Data Processing - Interactions



1. **Ingestion Hub** - The red ingestion box represents the possible ingestion mechanisms and components as previously described in Section "Ingestion". Within the context of big data processing to address high volume and high velocity data ingestion, a distributed file system or NoSQL database is commonly utilized. NoSQL databases are well suited for dynamic data structures and are highly scalable. The data stored in a NoSQL database is typically of a high variety because the systems are intended to simply capture all data without categorizing and parsing the data.

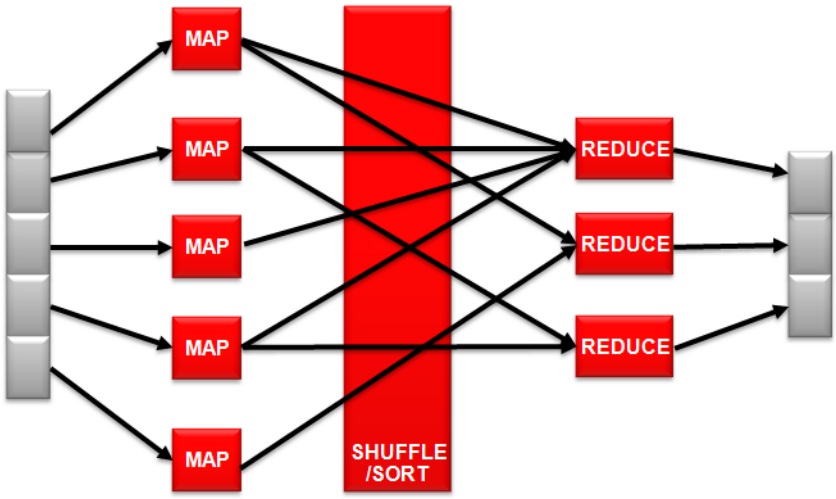
2. **Parallel Batch Processing Engine** - As previously highlighted, the current popular choice for parallel batch data processing is a programming model called MapReduce, which addresses the challenges arising from the scale of the data (e.g. volume, velocity), the varied formats (unstructured and semi-structured), and the complex techniques needed to process it.

   Distributed MapReduce tends to sit on top of distributed file systems. This allows MapReduce to distribute compute tasks across multiple nodes to where the data that needs to be processed is situated (best effort). Each node then processes the data stored on that node. This tends to alleviate the necessity to move large data sets over the network.

As highlighted in Section "NoSQL", just like many NoSQL systems, MapReduce utilizes the key/value pair as its data model. A MapReduce data processing task takes a set of input key/value pairs, and produces a set of output key/value pairs. One round of the data processing is generally divided into three phases: Map, Shuffle, and Reduce.

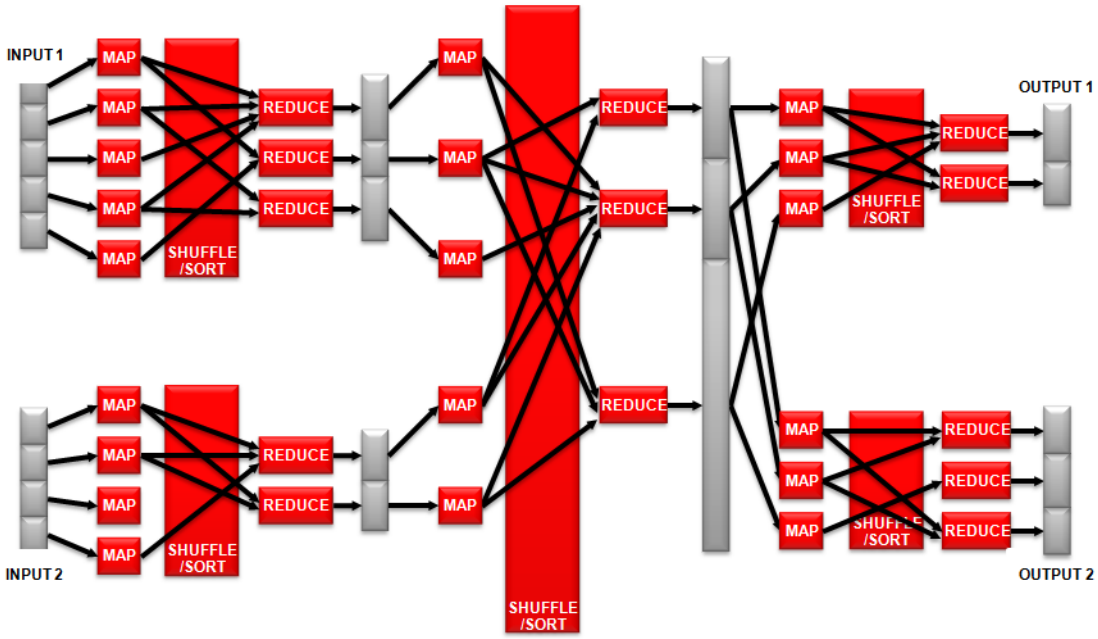
- **Map** - The Map phase breaks a job into small parts by executing a user defined mapper method to parse, transform, and filter input pairs and produce a set of intermediate pairs. Before the data is passed to the Reduce phase it needs to be sorted.
- **Shuffle** - The Shuffle phase, groups all intermediate values associated with the same intermediate key together, so they are ready to be passed to the Reduce phase.
- **Reduce** - The Reduce phase executes the user defined reducer method (e.g. count, average, min/max) to process the intermediate values associated with each distinct intermediate key. All values associated with a particular intermediate key are guaranteed to go to the same reducer. The reduce outputs zero or more final key/value pairs which are written to the distributed file system (See Figure 6–23).

Figure 6–23 Simple MapReduce Job



A complex MapReduce job might consist of a series of Map, Shuffle, Reduce rounds as shown in Figure 6–24.

Figure 6–24 More Complex MapReduce Job



Taking into consideration the characteristics of Big Data, the follow points assist in deciding if a parallel batch processing engine such as MapReduce is appropriate:

- Conventional data processing engines may be deemed too slow, expensive or not flexible enough to obtain the same results.
- The access and latency requirements allow for the batch processing of the data.
- Data processing required across structured, multi-structured, and unstructured data.
- The data to be processed has little or no data dependence. This allows the data located on one node to be primarily processed by the same node.
- The processing required has little or no computation dependence. This allows processing to primarily occur in parallel rather than waiting for one processing task to complete before the next processing task starts.
- No need for stringent transactional support, data integrity, and complex rule validations.

3. **Event Processing Engine** - The Event Processing Engine decouples the detection of events from the processing of events from the distribution of events. This has many advantages, for example, this enables the Event Processing Engine to continue detecting and capturing events even if the processing of previously captured events has not completed.

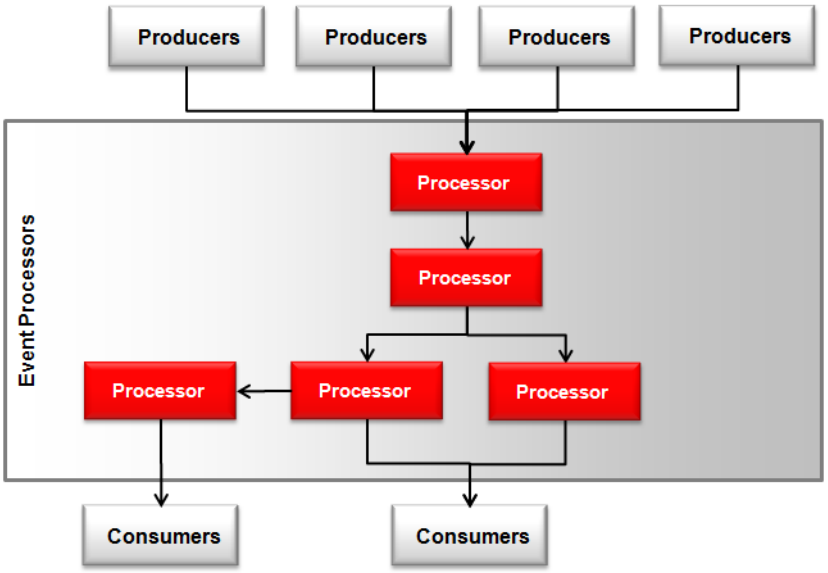
   Event producers, (both internal and external), are systems/components that generate events when a change occurs. For example, infrastructure services publishing alerts, or sensor devices that produce events based on physical changes such as goods movement. At the point of origination, the events may not be in a form suitable for consumption by the Event Processing Engine, (e.g. protocol and format). Therefore, an event handler mediates the differences. Event handlers support common protocols such as HTTP and JMS, but can be extended to support additional protocols and data formats (e.g. commercial twitter feeds).

Once complete, the event data is passed onto a channel. Channels are conduits that are responsible for queuing event data until the processor can act upon it.

Processing is the manipulation of event data through the application of business rules to identify opportunities, threats, and anomalies that need to be acted upon. Rather than applying algorithms against data, the processor should be seen as streaming event data through algorithms. The processor removes the event data from the channel to process it using rules predefined in the form of queries (e.g. Continuous Query Language - CQL). Capabilities such as pattern recognition, splitting, filtering, correlation, aggregation, and transformation are utilized to eventually generate the outbound stream of events.

Processing of events is rarely a one-step process. Real world use cases require the events to be processed a number of times applying the concepts of filtering, correlation, aggregation, etc. with different input combinations. The components of the Event Processing Engine can be arranged and connected together in a network fashion to perform a sequence of processing steps to address different purposes (See Figure 6–25).

Figure 6–25 Event Processing Network



When processing the inbound events, it is sometimes necessary to also use the contextual or historical information from the internal sources (e.g. NoSQL, HDFS, relational databases) to enhance detection and processing. For example, in the financial industry, the credit card processing events may be combined with the problem cards information from the internal sources to generate fraud detection events.

Once an actionable event has been detected, it is sent to the outbound channel. Outbound channels are processing endpoints that receive the output from the processors. These events need to be distributed to the interested consumers for either further processing or stored accordingly (e.g. relational database, NoSQL database). The role of the event distributor is the reverse of the role of the event handler The Event distributor distributes the outbound events using the appropriate protocol. For example, the event distributor can be used to publish events to a JMS queue or topic. This is especially useful when consumers want to receive offline events as they can utilize a JMS topic durable subscription.

While the Event Processing Engine can perform a level of Big Data Processing, (e.g. early alerting), it is common for additional and more comprehensive processing to also take place on a less time-sensitive basis (e.g. using a Parallel Batch Processing Engine). The use of an Event Processing Engine as part of Big Data Processing is to immediately filter out irrelevancies and identify patterns of interest that can be used to an organization's advantage or to defuse an immediate business threat.

For more details regarding Event Driven Architectures refer to the *ITSO EDA documents*.

4. **Exploratory Processing Engine** - As mentioned earlier, the Exploratory Processing engine utilizes a hybrid, search/analytical database that provide an unprecedented flexibility in combining diverse and changing data as well as extreme performance in analyzing that data. It supports the navigation, search and analysis of any kind of information including structured, semi-structured, and unstructured content.

   The database employs a flexible data model that reduces the need for up-front modeling and enables the integration of diverse and changing data. The database does not employ an overarching schema for data, instead every record is its own schema. This enables the rapid integration of any data type, whether structured or unstructured, from inside or beyond the confines of the Data Warehouse, without the efforts associated with traditional relational data modeling.

   The Exploratory Processing Engine in conjunction with a data integration engine performs a number of ingestion and enrichment activities that transform, standardize, and process data from various sources (e.g. structured operational data, unstructured content) for the purpose of supporting exploratory analytics.

   As this varied data is ingested it can be stored, enriched, and organized. Data is unified via derived common attributes from attached metadata. For example, structured data may be ingested from a Data Warehouse. The attributes are derived from the column names of the table. During bulk/incremental ingestions, the records are mapped to a classification taxonomy, which enables exploratory analytical tools to access the records and allows the users to search, refine, and navigate the records. Each record consists of an arbitrary collection of attributes made up of key/value pairs. These attributes can have a single value, multiple values, or a hierarchy of values.

   a. The unstructured content, such as chatter on social media channels, product reviews from the web, business documents in content management systems, and text fields in a Data Warehouse, needs to be pre-processed before it can be ingested by an Exploratory Processing Engine. Leveraging text analytics and natural language processing, (e.g. parsing, cleansing, tokenizing), a Content Acquisition Engine can extract new facts and entities such as people, location, and sentiment from text that can be used as part of the analytic experience.

      The Content Acquisition Engine performs filtered, full, and incremental crawls of data sources (e.g. file systems, content management systems, web servers). The Content Acquisition Engine utilizes a document converter that enables the conversion of crawled binary content (such as Microsoft Word documents and Adobe Acrobat PDF files) into text. The crawled data is then intelligently tagged with metadata properties that are derived from the source documents. The Content Acquisition Engine then stores the resulting output into a generational database. The Exploratory Processing Engine can then ingest data from a generational content store, which has been pre-processed and populated via the use of a Content Acquisition Engine.

Refer to the *ITSO Business Analytics* documents for more details regarding Exploratory Analytics.

6.3.3 Enterprise Master Data Management

Master data represents the key business entities that support operational data such as customer, product, financial structures and sites. In addition to supporting operational data, master data also supports analytical data by representing key dimensions on which analytics is accomplished. Master data are, in effect, the trusted, quality, single version of the truth for key business entities that an enterprise requires.

Maximum business value comes from managing both operational and analytical master data across multiple domains. These solutions are called Enterprise Master Data Management.

6.3.3.1 Operational Master Data Management

Master Data Management (MDM) solutions that focus on managing operational data to ensure quality and consistent access for use by operational applications are called Operational Master Data Management.

Operational MDM improves the operational efficiencies of the applications themselves, and the business processes that use these applications, by relying heavily on data ingestion, data virtualization, and information governance technologies.

They bring real value to the enterprise, but lack the ability to influence reporting and analytics.

6.3.3.2 Analytical Master Data Management

Analytical Master Data Management focuses on managing master data to support downstream consumption in the delivery of consistent historical and analytical reporting. One common approach is for Analytical MDM to manage master data items and associated hierarchies that are required for aggregation and analysis reporting.

Analytical MDM brings real value to the enterprise, but lacks the ability to influence operational systems. Any data cleansing done inside an Analytical MDM solution is invisible to the operational applications. Because Analytical MDM systems do nothing to improve the quality of the data under the operational application landscape, poor quality, inconsistent authoritative data finds its way into the analytical systems and drives less than optimum results for reporting and decision making.

Operational MDM without Analytical MDM only solves half the master data problem. Analytical MDM without Operational MDM does not have the necessary dimension data to achieve its promise. Multi-Entity Enterprise Master Data Management creates a single version of the truth about every master data entity. This data feeds all operational and analytical systems across the enterprise.

6.3.3.3 Multi-Entity Master Data Management

Multi-entity Master Data Management solutions use an enterprise authoritative data schema with a common set of services coupled with a combination of applications and technologies that consolidates, cleans, and augments this enterprise master data, and synchronizes it with all applications, business processes, and analytical tools. This results in significant improvements in operational efficiency, reporting, and fact based decision-making. Example entities that can be leveraged across all functional departments and analytical systems include:

- **Customer** - Also known as customer data integration (CDI) that enables organizations to centralize information from heterogeneous systems, creating a single enterprise-wide 360-degree view of customer information.
- **Site** - Enables organizations to centralize site and location specific information from heterogeneous systems, creating a single view of site information.
- **Supplier** - Unifies and shares critical information about an organization's supply base. It does this by enabling customers to centralize all supplier information from heterogeneous systems and thus creates a single view of supplier information.
- **Product** - Also known as product information management (PIM) enables organizations to centralize all product information from heterogeneous systems, creating a single view of product information and eliminating product data fragmentation.
- **Financial** - Enables an enterprise view of financial chart of accounts, cost centers, and legal entities with a view to govern on-going financial management and consolidation based on consistent definitions of financial and reporting structures across general ledger systems, financial consolidation, planning and budgeting systems.
- **Relational Management** - Creates an enterprise view of analytical dimensions, reporting structures, performance measures, and their related attributes and hierarchies.

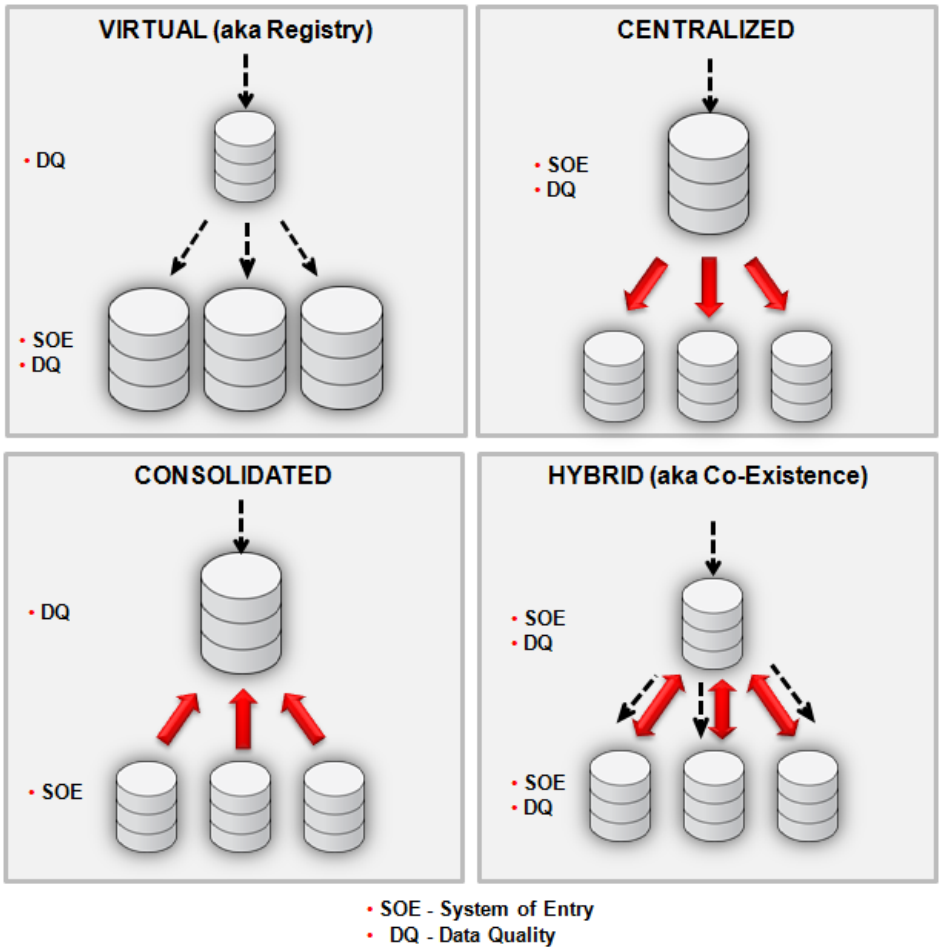
Incremental deployment of MDM tends to focus on one authoritative entity. More entities can be added over time. But, it makes sense to make sure that these MDM entities do not become data silos of their own. Therefore, Enterprise MDM should take a holistic view of these individual authoritative entitles.

6.3.3.4 Master Data Management Patterns

It is often said that Master Data Management is a set of disciplines, processes and technologies. This is true, as without an information governance program in place a technology-only approach to master data management will fail to deliver business benefits. An information governance program is out of scope for this document, and as such this section will highlight the underlying information provisioning architecture for master data management.

There are a number of architectural approaches and patterns to provision a quality single version of the truth with respect to key business entities and reference entities. The most common is frequently referred to as a hub design due to its central focus of access, but not necessarily the same system of entry (SoE). See Figure 6–26 for the most common hub designs.

Figure 6–26 Common Hub Designs



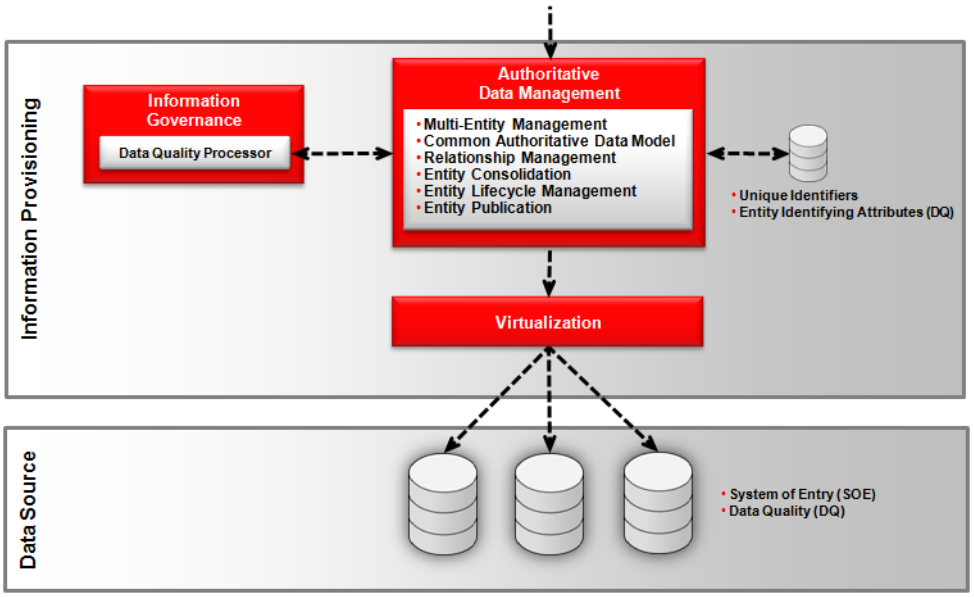
Each of these hub designs has its own architecture characteristics, advantages, and disadvantages. The architect must decide which hub design is appropriate, taking into consideration the current challenges, existing systems, funding, and commitment.

Invariably, the use of more than one hub design is the most likely course of action, either to address different requirements, or as part of a transition program. These hub designs require an initial load of data to lay a solid foundation of quality data. Afterward, the capabilities of the hub keep the data synchronized.

6.3.3.4.1 Virtual Hub

A virtual hub (aka registry hub) leaves the authoritative data in the source systems and uses data virtualization techniques to dynamically assemble a master record in real-time (See Figure 6–27).

Figure 6–27 Virtual Hub



This virtualization approach requires the virtual hub to store a master index of unique identifiers which map to source system key references to assist in the retrieval of the master record. In addition to the unique identifiers and source system key references, the virtual hub will store entity-identifying attributes to assist with search requests from information consumers without having to go the source systems after every search request. The number of identifying attributes is variable depending on identification requirements.

Each source system remains in control of its own data (SOE) and there is a reliance on data quality being applied by the source systems. But the identifying attributes held within the hub should be cleansed.

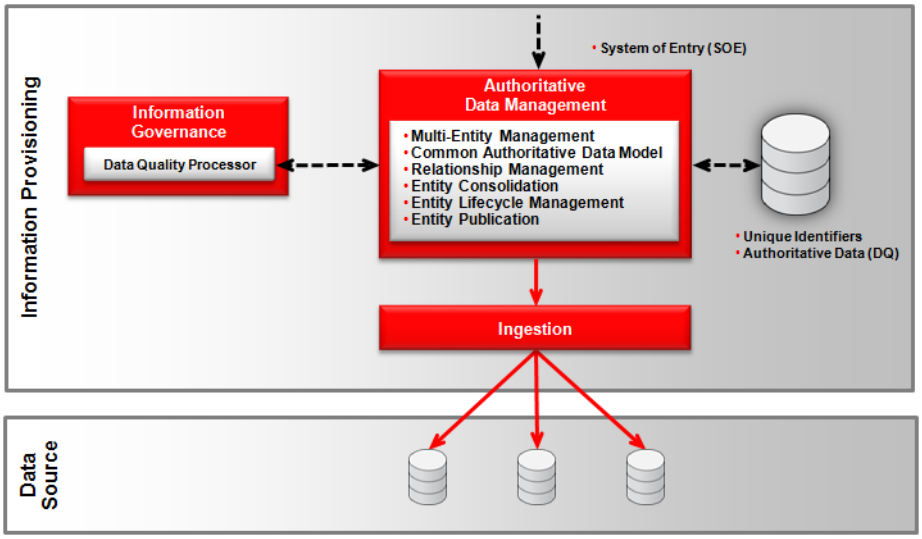
Due to the complexity and capabilities of the source systems, a virtual hub tends to be read-only where the survivorship rules are applied at data access.

While this hub design might seem ideal due to having no impact on existing applications to be updated, it does have a heavy reliance on the source systems for both performance and reliability.

6.3.3.4.2 Centralized Hub

A centralized hub (aka transaction hub) stores all authoritative data for an entity and also is the system of entry (SOE). The centralized hub can be seen as the opposite of a virtual hub where the data and system of entry are de-centralized (See Figure 6–28).

Figure 6–28 Centralized Hub



The centralized hub stores all authoritative data, unique identifiers and any new attributes which are not found in the underlying operational systems.

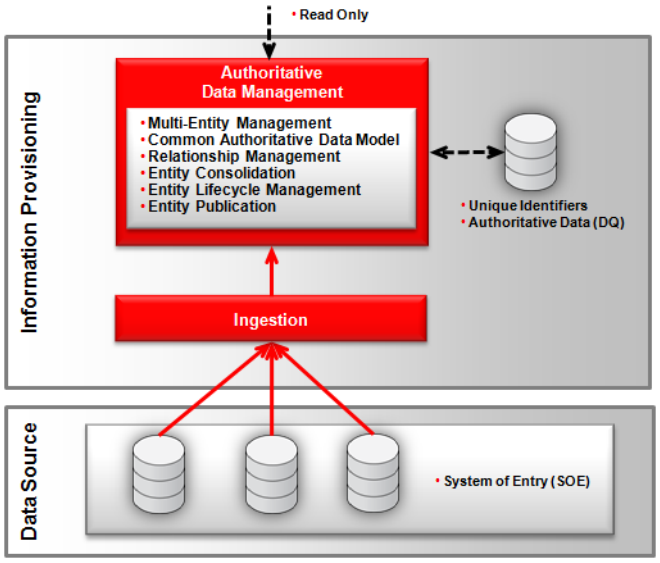
This hub manages all entity lifecycle operations (Create, Update, etc.) and interacts with a data quality processor to make sure that data has been cleansed. The centralized hub is a read/write system and has a large impact on applications that need to be updated so that any changes to authoritative data are made through the hub as the operational systems are not the system of entry anymore. This, in effect, makes this hub the 'System of Record' (SOR).

If required, updates to the authoritative data in the centralized hub can be published and propagated to subscribing systems via data ingestion techniques such as bulk data movement and incremental data movement.

6.3.3.4.3 Consolidated Hub

A consolidated hub aggregates authoritative data from source systems using ingestion techniques and stores it within a master data store in the consolidated hub (See Figure 6–29).

Figure 6–29 Consolidated Hub



This consolidated approach requires the storage of unique identifiers and of quality authoritative data. Each source system remains in control of its own data (SOE) but the consolidation technique ensures that both data quality processes and survivorship rules are applied to the authoritative data at the time of consolidation.

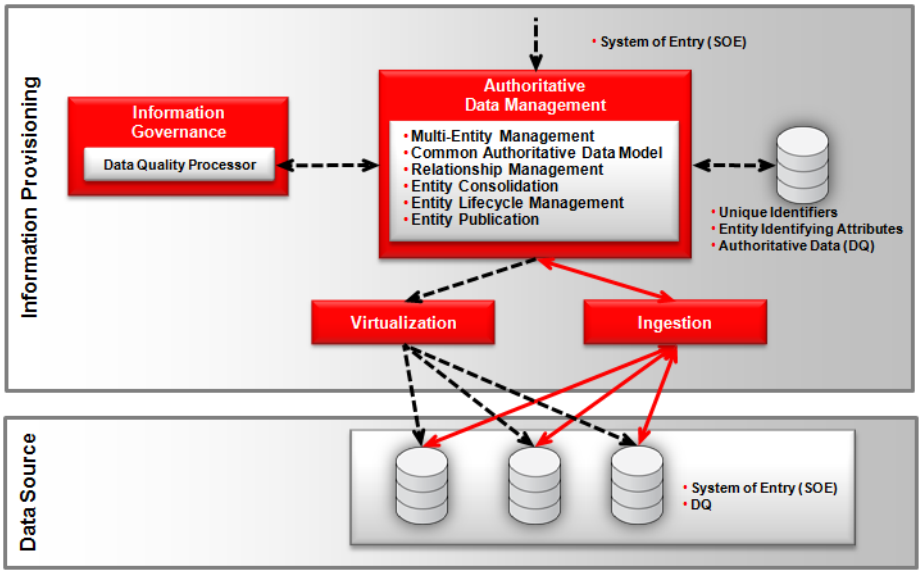
Like the virtual hub, the consolidated hub is read-only and has no impact on existing applications to be updated.

The consolidated hub is less reliant on the source system than the virtual hub as performance and reliability of the source systems are hidden to some extent due to the data being available for consumption from within the consolidation hub. But due to the consolidated nature of this hub, the data within the hub can become stale due to the latency between consolidations.

6.3.3.4.4 Hybrid Hub

A hybrid hub (aka co-existence hub), as indicated by the name, is a combination of the capabilities of a virtual hub, consolidated hub and centralized hub (See Figure 6–30).

Figure 6–30 Hybrid Hub



Authoritative data are stored in both the hub and in the source systems. The hybrid hub is commonly used as a transition architecture when moving from a virtual hub to a centralized hub.

During this transition there will invariably be a mixture of applications that utilize the hub as the primary source of authoritative data and applications that do not. Applications that do not use the hub as their primary source of authoritative data are still the system of entry for their specific systems. The hub can also be a system of entry where changes to the authoritative data can be propagated to subscribing applications. This hybrid approach leads to an increased level of complexity over a virtual hub due to the synchronization that is required between the hub and the source systems, which may highlight a number of discrepancies and conflicts that will have to be managed.

Another approach to employing a hybrid hub is more in line with the virtual hub but attempts to address the performance issues related to virtualization techniques. As with the virtual hub, the hybrid hub contains unique identifiers and mapped source system key references. It also contains the consolidated authoritative data from the source systems of the most common or the most important attributes. This allows the hybrid hub to respond to information consumer requests by using authoritative data stored in the hub. When it cannot be satisfied by replicated data alone, then the hybrid hub utilizes the same virtualization approach described in the virtual hub. As with any form of replication there is data latency which will lead to the hybrid hub containing stale data.

The hybrid hub is responsible for applying data quality processes to the replicated data and management of entity-identifying attributes. But it also relies on the source systems to cleanse its data when virtualization access is utilized.

Both performance and reliability are dependent on the source systems when utilizing virtualization access.

6.3.3.5 MDM-Aware Consumers

When an operational application understands that the key data elements within its domain have business value beyond its borders, they are "MDM-Aware". An MDM-Aware application is prepared to:

- Use outside data quality processes for data entry verification
- Pull key data elements from an outsaide master data source
- Push its own data to external maser data management systems

In effect, MDM-Aware applications have the ability to synchronize data to MDM Hubs, fetch master data from MDM Hubs, and utilize MDM data quality processes at all data capture points.

Making an existing operational system MDM-Aware entails changing its behavior such that it can operate with the master data sitting outside of its database, without impacting support of its core processes and functions.

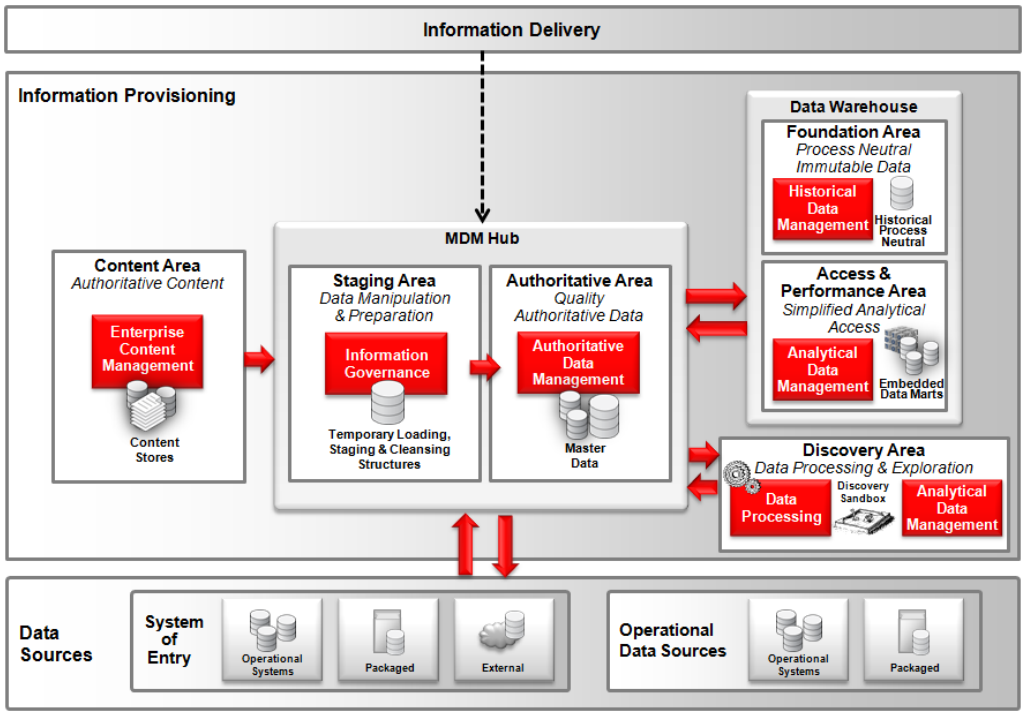
6.3.3.6 Illustrative Example

With the various MDM requirements, MDM patterns and information acquisition techniques available, there is no single way to architect MDM. For illustrative purposes the rest of this section will focus on an example scenario highlighting information acquisition mechanisms, components and interactions points, while utilizing a distributed system of entry approach. These are by no means the only interactions, but highlight the number of options and flexibility that a MDM solution can offer.

6.3.3.6.1 Areas of Responsibilities

Figure 6–31 illustrates the ingestion points between various areas of responsibility within the information provisioning layer. The MDM Hub is composed of two key areas of responsibilities. The Authoritative Area manages the master and key reference entities, which closely interacts with the staging area for the cleansing of ingested data. This ingested data can originate from a number of areas, including operational data sources, Data Warehouse, and the discovery area. In turn, the MDM Hub is also a provider of master data to a number of areas.

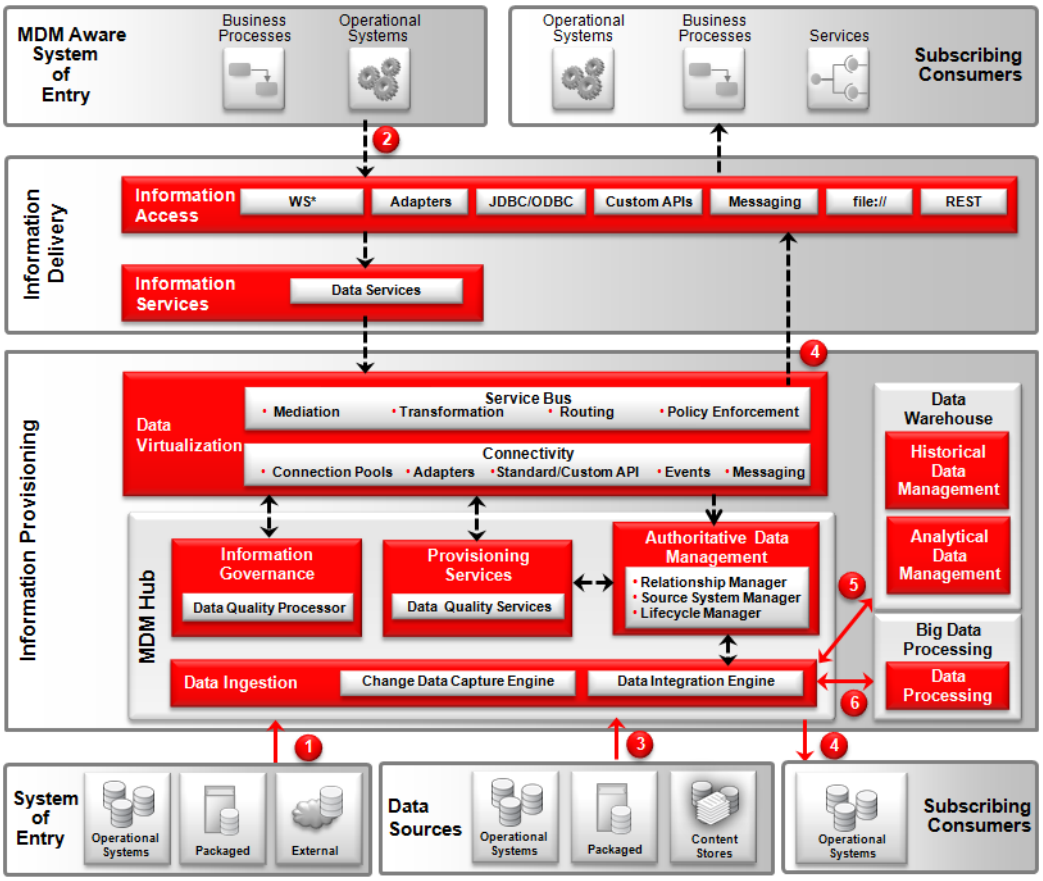
Figure 6–31 MDM - Areas of Responsibilities



6.3.3.6.2 Interactions

In addition to ingestion mechanisms Figure 6–32 highlights MDM making effective use of service oriented architecture to propagate and expose the master data to the interested applications. Without utilizing a service-oriented approach to MDM there is a danger that MDM creates quality master data that becomes its own data silo. In effect SOA and MDM need each other if the full potential of their respective capabilities are to be realized.

Figure 6–32 Example MDM Interactions



1. Sometimes it is not feasible to update existing operational systems to be MDM-aware. This scenario can occur when application source code is not available, proprietary legacy applications do not allow changes in the interfaces, or when there is a plan to decommission a system soon and therefore system's enhancements are not considered worthwhile. In this case operational systems that are utilized as a system of entry act as passive participating applications where data changes in these systems are detected and pushed to the MDM hub for cleansing, de-duplication and enrichment. This is in effect performing a synchronization of authoritative data.

   It's important to be able to dynamically synchronize data into and out of the hub. The synchronization doesn't have to be real-time, although there are benefits in doing so. Since the whole point is to build a "Single Source of Truth" for a particular entity like customers or products, having out-of-date information in the hub, or not synchronizing the data quality improvements back to the original source systems, can defeat the whole purpose of the project.

This synchronization can utilize many different approaches such as asynchronous or batch. This non-intrusive approach in the architecture can lead to quicker MDM implementations. As data flows from the operational system into the MDM hub it is transformed into the common authoritative data model. The MDM hub makes full use of the data quality processor via the appropriate Provisioning Services. This loose coupling approach to invoking data quality processes allows the MDM hub to utilize a number of different data quality processors depending on the entity in question, as well as to aid in the easy replacement of a data quality processor, if the need should arise in the future. The MDM hub consolidates, cleanses and enriches, de-duplicates and builds a golden record.

2. Where it is feasible to update existing operational systems to be made MDM-aware, a synchronous pull approach is utilized at the time of data entry. The operational system makes an authoritative data service query request for a list of records that match a specified selection criterion. The MDM hub searches its master index and returns the list of records. Armed with these records the operational system enables the end user to peruse and select the appropriate authoritative record. Once selected, the operational system makes another data service request to the MDM hub - this time to fetch the attributes for the selected record in question.

   These match and fetch data service requests are synchronous and enable a real-time interaction between the operational system and the MDM hub. Through these data service requests, consuming applications have on-demand access to the single source of truth for authoritative data, thus preventing duplicate and inaccurate data entry. Once the operational system has created a new record, and/or updated an existing record, the authoritative record needs to be synchronized with the MDM hub in the same manner as highlighted in bullet point 1 above.

3. No matter what MDM hub pattern is used, an initial load of authoritative data are required from the source systems into the MDM hub to build the initial golden records. Due to the volume of data this tends to be performed by a bulk data ingestion mechanism. From time to time delta loads may be necessary. This load can contain both structured and unstructured data. If the unstructured data are already contained within a content store with adequate security and versioning support, it is best to leave the unstructured content where it is and load the associated content store reference key into the MDM hub.

4. The MDM Hub's source system manager ensures that a central cross-reference is maintained enabling the MDM hub to publish new and updated records to all participating subscribing information consumers. The data are transformed from the common authoritative data model to the format needed by target systems. Before publishing the data, the MDM hub enforces any associated security rules making sure that only data that is allowed to be shared is published. A key benefit of a master data management solution is not only to clean the data, but to also share the data back to the source systems as well as other systems that need the information. It is preferable for the MDM hub to feedback the golden record back to the MDM source systems. Otherwise the result is a never-ending cycle of data errors. Therefore the source systems that initially loaded the data, or systems that took part of a delta load, may also be subscribing information consumers.

   An internal triggering mechanism creates and deploys change information to all subscribing consumers using a variety of mechanisms such as bulk movement and messaging. All changes to the master data in the MDM Hub can also be exposed to an event processing engine. The hub automatically triggers events and delivers predefined XML payload packages appropriate for the event. This enables the real

time enterprise and keeps master data in sync with all constituents across the IT landscape. See *ORA EDA Foundation* for more information.

5. The MDM hub holds accurate, authoritative, governed dimension data, the actual operational data cross-reference, and the hierarchy information on all key master data entities. With all the master data in the MDM Hubs, only one pipe is needed into historical/analytical data management systems (such as a Data Warehouse) to provide key dimensions such as Customer, Supplier, Account, Site, and Product.
6. The MDM hub holds accurate, authoritative, and governed data that can give contextual information to data processing activities. For example, authoritative data such as master IDs which can enrich event processing in identifying, aggregating, and publishing events. In addition, external data sets such as social media and government data coupled with insights achieved through big data processing can embellish data held within a MDM hub. For example, utilizing identity resolution mechanisms the MDM Hub will be able to link new sources of external data such as social metrics with key master data such as customers. This will enable insights such as customer and product sentiment, identifying previously unknown relationships, and organization hierarchy.

6.3.4 Data Warehouse

A Data Warehouse is a specially prepared repository of data designed to support decision making by enabling reporting and analytics. The Data Warehouse can be seen as a primary storehouse of cleansed, transformed, and consolidated historical data for an organization that provides business users with a time-based, integrated view of cross-functional data.

To create the Data Warehouse, data are extracted from source systems (i.e. operational systems and external sources), cleansed, transformed and summarized, and loaded into the Data Warehouse. As defined by Bill Inmon[2], the data stored in a Data Warehouse have the following characteristics:

- **Subject Oriented** - The data relating to the same event or object are logically organized e.g., customers, sales.
- **Time Variant** - Changes to the data are tracked, recorded, and maintained leading to a store of historical data.
- **Nonvolatile** - Data are read only, and are never overwritten or deleted, but retained for future reporting.
- **Integrated** - Data from many different sources are combined and analyzed for consistent results.

As previously mentioned, a Data Warehouse draws data from operational systems, but is physically separate and serves a different purpose. Operational systems have their own databases and are used for transaction processing. Commonly known as Online Transaction Processing (OLTP), they support transaction oriented applications for efficient real time querying and transaction processing (i.e. Inserts, Updates, and Deletes). Whereby a Data Warehouse has its own database and is used to support decision making by enabling reporting and analytics. Once the warehouse is created, users access the data in the warehouse using a variety of tools and applications.

A Data Warehouse is not the place to solve queries such as "Did this invoice get paid?" but rather questions such as "What were the 20 most efficient Hospitals for procedure X in state Y?"

[2] Building the Data Warehouse by William Inmon (John Wiley & Sons)

The warehouse is not constrained by business process or OLTP system boundaries but should be seen as a complement to OLTP systems. Since it may be used for purposes other than analytics, the warehouse may be modeled in a business-normalized form as opposed to a dimensional form.

6.3.4.1 Active Data Warehouse

Initially data was loaded into a Data Warehouse on a fixed batch schedule (e.g. monthly basis to support monthly reports). But now data are scheduled to be loaded into the Data Warehouse during off peak periods, usually in the middle of the night. But increasing data volumes, decreasing batch windows, and management dashboards requiring data fresher than 24 hours, have increased the pressure on organizations to move towards more frequent intraday batches.

In addition to the traditional Data Warehouse approach, active data warehousing has emerged. An Active Data Warehouse extends a traditional Data Warehouse by loading and maintaining the Data Warehouse in an extremely up-to-date mode. Instead of the Data Warehouse being updated once or twice a day, it is being updated in seconds. This allows it to support operational requirements and real-time processes and removes the challenges of ever decreasing batch windows.

An Active Data Warehouse makes use of real-time feeds which are critical to ensure that the business reports are not looking at stale week-old data. The more current the data, the better and more informed the decision making will be. In many such scenarios, change data capture (CDC) plays a key role in keeping data consistently updated without impacting the target or source performance.

6.3.4.2 Areas of Responsibility

With the inevitable evolution of an organization's Data Warehouse architecture, it is important to define distinct areas of responsibility within the Information Provisioning Layer. These distinct areas not only have distinct capabilities but also have their own logical data stores to support their capabilities and the part they play in an overall Data Warehouse approach.

The following section highlights the characteristics of each area within the context of Data Warehousing. For more details refer to Section "Areas of Responsibility".

6.3.4.2.1 Staging Area

The Staging Area can act as either a temporary and/or permanent area where data manipulation, cleansing and the production of quality data are performed. The Staging Layer is the first destination of data that has been acquired and provides a level of isolation between data that will be moved into the Data Warehouse and data that is generally available to consumers.

Data can be acquired at various rates, (e.g. in small frequent increments or large bulk transfers), asynchronous to the rate at which data are refreshed for consumption. The rate and frequency at which data are acquired, and the rate and frequency at which data are refreshed in the Data Warehouse, is driven by business needs.

While many of the old rules regarding the static qualities of a Data Warehouse have now gone, it is still true that the Data Warehouse must contain data that is clean, consistent, and complete, as far as is practical. The Staging Area is where business rules are applied to achieve these objectives. The Staging Area is where capabilities of data quality management are applied in order to achieve clean, consistent, and complete data. Once cleansed, data can then be moved into the Foundation Area or Access and Performance Area depending on which architecture pattern is employed.

It is imperative that any movement of data does not in itself generate data quality issues, as this will produce faulty reporting and misleading analytics.

6.3.4.2.2 Foundation Area

The Foundation Area is responsible for managing data for the long term. It may also be called the Atomic Data Area since it maintains data down to the lowest level of granularity.

It is in the Foundation Area that data from all originating sources is maintained as a unified schema. Although actual consolidation may take place during data ingestion processes or staging, the master version resides here.

In order to most easily adapt to changes in the organization over time, data are maintained here in a business neutral form. Changes to organizational structures and dimensions should not impact the way data are structured in this area. Versatility takes precedence over navigation and performance. Likewise, in order to efficiently store large volumes of data, this area is normalized in a manner such as 3NF.

6.3.4.2.3 Access and Performance Area

The Access and Performance Area is used to represent data in ways that best serve the analytical community. Since it does not have responsibility for historical data management, it is free to represent data for most efficient access and ease of navigation.

Often the Access and Performance Area will consist of dimensional models and/or cubes. In addition, it may contain aggregations of facts (rollups) for rapid query responses. Access and Performance Area structures can be instantiated and changed at will in order to conform to the business hierarchies of the day.

6.3.4.2.4 Authoritative Area

The Authoritative Area is responsible for the master and reference entities that can support an organization's reporting and analytical needs. The Authoritative Area can supply critically important dimensions, hierarchies, and cross reference information to the Data Warehouse. With all the master data in the Authoritative Area, only one pipe is needed into the Data Warehouse for key dimensions such as Customer, Supplier, Account, Site, and Product.

The Authoritative Area maintains master cross-references for every master business object and every attached system. This cross-reference is available regardless of which data warehousing architecture style is employed. This enables the seamless combination of historic data retrieved from those operational systems and placed into a fact table, with real-time dimensions form the Authoritative Area Hub. This way analytical operations derived from the Data Warehouse is based on the same data that is used to run the business on the operational side.

6.3.4.2.5 Content Area

While the majority of the data sources for the Data Warehouse are structured in nature, increasingly there is also a need to supplement this with unstructured data. The Content Area is responsible for providing access to unstructured data in the format required by various information consumers.

Data are typically either used for simple reference purposes - for example a Marketing Analyst may want to see the marketing collateral used for a highly responsive segment of customers - or to enrich the structured data through further processing using Text Mining or classification, clustering, or feature extraction techniques.

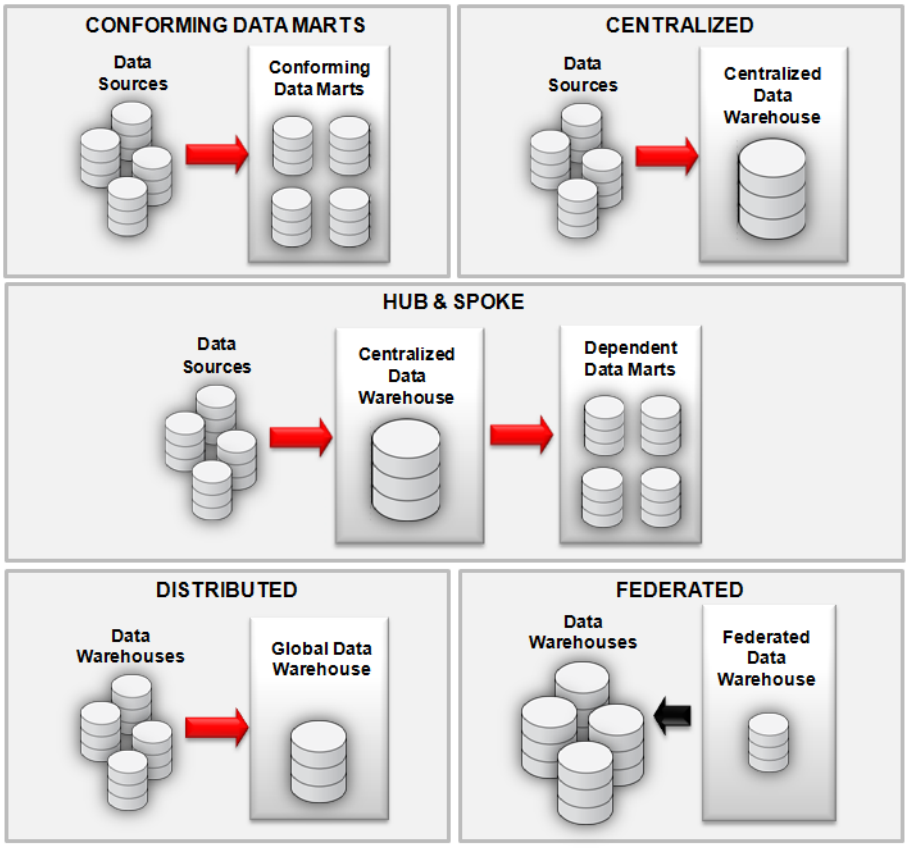
6.3.4.3 Data Warehouse Patterns

To deliver a Data Warehouse that meets the needs of the organization requires not only an architecture, but also a set of disciplines, processes, and technologies. For example, approaching the development of a Data Warehouse without an information governance program in place and taking a technology-only approach will fail to deliver business benefits. An information governance program is out of scope for this document and as such this section will highlight the underlying information provisioning architecture for a Data Warehouse.

If a business unit is left without guidance they will tend to develop their own data repositories. These data repositories will invariable be independent of other data repositories. While this approach might address the immediate needs of the business unit in question, it will invariably lead to costly integration efforts later on.

There are a number of architectural approaches and patterns to provision a quality Data Warehouse. See Figure 6–33 for some example patterns.

Figure 6–33 Example Data Warehouse Patterns



Each of these Data Warehouse patterns has their own architecture characteristics, approaches, and merits. The architect must decide which pattern are appropriate, taking into consideration the current challenges, existing systems, funding, and commitment.

While there has been much debate over which architecture is best, one thing for sure is that the design of a Data Warehouse will change and evolve over the years. This change is due in part to technological advancements (which lower the cost of resources), changing business user analytical requirements, increasing scope and

domains, and the maturing of the enterprise as the Data Warehouse becomes the foundation for reporting and analytical needs.

While Figure 6–33 is not an exhaustive list of Data Warehouse patterns, these foundational and architecturally significant patterns enable an architect to define additional multifaceted patterns by combining two or more patterns together into a hybrid architecture.

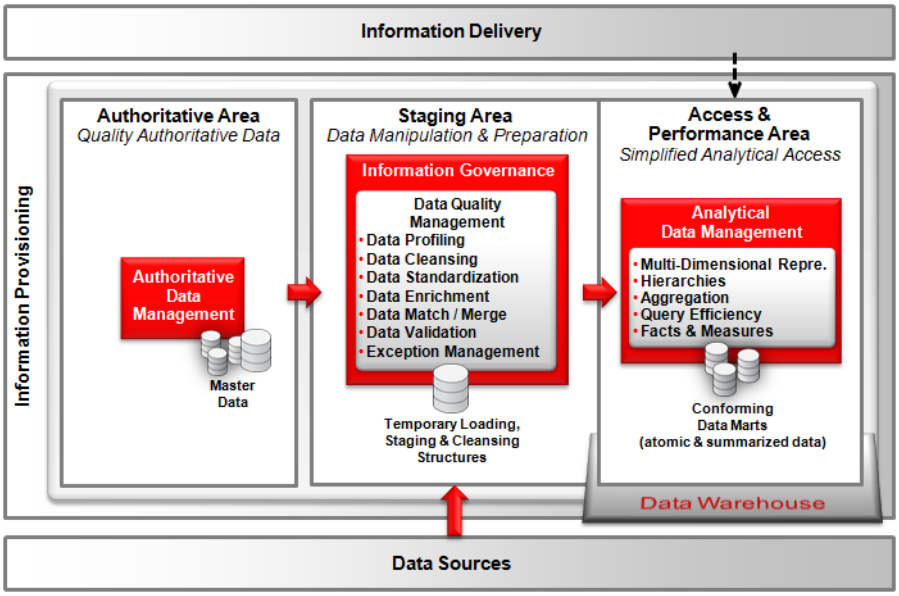
More details regarding these architectural patterns will follow later in the document. As a reminder, this paper focuses on the architecture aspects and does not cover any details on methodology associated with these patterns.

6.3.4.3.1 Conforming Data Marts

A conforming Data Mart approach involves the use of Data Marts and dimensional models. This approach is intended to produce results quickly by dividing the scope of the warehouse effort into several subject areas. Each subject area can relate to a business process, such as sales, support, operations, etc.

Data Marts can be provisioned and maintained at departmental levels, as opposed to creating a single enterprise-wide Data Warehouse. This reduces the effort required to prioritize, approve, and fund the project and allows each department to advance their program at their own pace.

Figure 6–34 Conforming Data Marts



Data sources pertaining to a subject area are collected from operational systems, (and if available authoritative data), into a temporary Staging Area for cleansing and transformation into a dimensional format. This temporary Staging Area can be seen as a holding area for data that needs to be captured, integrated, consolidated, and transformed. It provides cleansed, quality data which are loaded into the Data Marts.

The data in the temporary Staging Area is often in the form of a star dimensional format. Data are then ready to be transferred to the Data Marts, after which the data in the temporary Staging Area is purged. Since both the Staging and Access and Performance Areas are in a star dimensional format, the movement requires little data conversion. Often the data movement involves an UPSERT process. An UPSERT

process will update the row in the Data Mart if it already exists, or insert the row if it does not already exist.

The Data Marts have atomic (detailed) level data and summary data. There is no single Data Warehouse, but adopters of this architecture state that the accumulation of all of the Data Marts constitutes a Data Warehouse. With the appropriate governance in place the Data Marts can exhibit common elements called conformed dimensions.

Benefits

Pressure from the business demands results quickly and in a cost effective manner. The conforming Data Marts approach can be implemented incrementally, which enables the delivery of the first results relatively quickly. In addition, taking an incremental approach allows organizations to learn and increase their expertise by understanding what works, and more importantly, what does not work within their environment. This enables future Data Marts to avoid the same mistakes.

This popular approach is the dimensional approach that simplifies the data model to facilitate access to the data for querying by end users using stars or snowflakes. Very much a physical model, drill paths, hierarchy and query profile are embedded in the data model itself rather than the data, and in part at least, this is what makes navigation of the model so straightforward for end users.

Dimensional models place emphasis on information access and not the data management part of Data Warehousing. They simplify user access along well known paths but there are some forms of problems that are not well served by the techniques embodied in this approach, requiring additions such as lookups, helper tables, 'fact-less' fact tables and the like. The approach simplifies access but may limit the depth of analysis possible. This may of course not be an issue for some businesses but will be for others.

The use of dimensional models makes data easily consumable by a wide variety of tools and applications. Tools are generally capable of interpreting the dimensions and provide a platform for analysis without the need for software programming and DBA expertise. In addition, when data are easily accessible, understandable, and provide high performance, users perceive them to be consistent and well integrated.

By employing temporary data stores, this architecture can be less resource intensive and therefore more cost effective. This approach builds the Data Warehouse structure to maximize usage efficiency while minimizing the scope of data collection.

Challenges

With taking a temporary staging approach, the data acquisition and movement between areas tends to be more difficult, less flexible, and sometimes can lead to short-term gains at the expense of a long term architectural success.

Since the analytical needs of an organization may span multiple subject areas, the architecture must provide the ability to perform queries across Data Marts. Therefore it is critically important to ensure that data are compatible across subject area data models.

Conformed dimensions are required for the success of this approach. One of the real challenges for a dimensional-only approach is in the development of conforming dimensions across the business. This is a nontrivial activity to achieve if using an iterative development approach.

If an organization cannot justify a strategic approach to building a Data Warehouse, then there may be some concerns that the time and budget may not be available to apply information governance to these Data Marts to make sure that their dimensions conform. Without proper information governance the Data Marts will be little more

than a collection of independent Data Mart silos with redundant data and inconsistent results across Data Marts.

Unfortunately, since data are being managed by separate Data Mart DBMS systems, conformance is often a matter of proper governance, (people and process), as opposed to being strictly enforced by technology. For example, in a healthcare model, there would be a common definition and agreement on what a patient is, and the attributes that make up the patient table, (e.g. Patient Id will be number with a maximum of 20 digits). This agreement is critical and can be difficult to get among competing departments. Because of this challenge, and taking an incremental approach, it is not uncommon for the initial Data Marts to have incorrect definitions that will need to be corrected. It is common for end users not to want to relinquish control or access over their Data Marts in fear that they might need the data at a future time and it will not be available.

In addition to the challenges of defining, implementing, and maintaining conformed dimensions across disparate systems, this approach has certain drawbacks due to its heavy reliance on dimensional models. Dimensional models are well suited to analytical navigation, but are not the most effective way to model business data for historical purposes. For instance, as changes occur to the business over time, dimensions will need to change, and likewise business entities and their attributes will also change. The dimensional model can remain clean and simple if it represents either the current hierarchical state or a previous state, but can become quite convoluted if it tries to represent different versions of hierarchies over time.

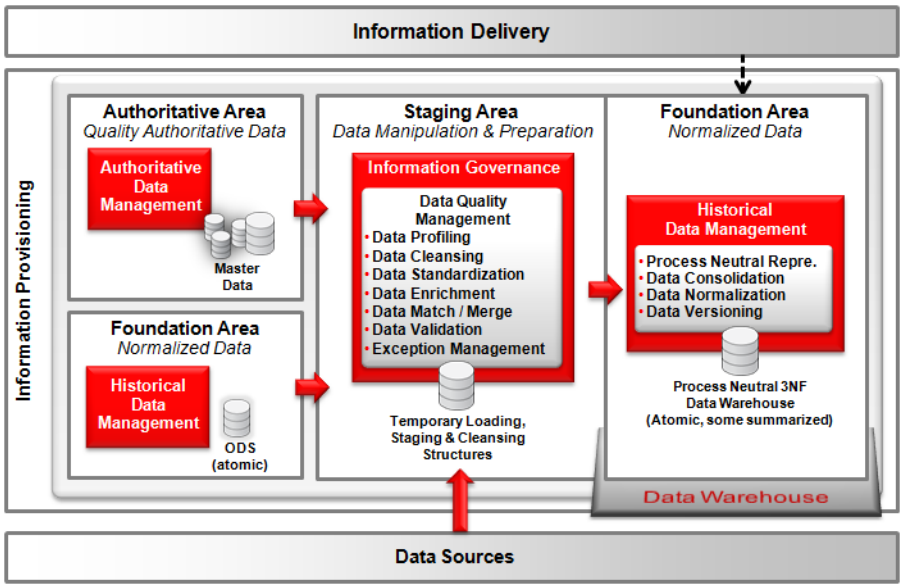
Lastly, it is very common for organizations to have requirements which will necessitate the definition and creation of additional data structures to support operational and historical reporting and data mining

6.3.4.3.2 Centralized

As the name suggests, all data whether it is atomic or aggregated, resides within a central store, with users directly accessing this central store with the relevant tools and applications.

The centralized approach is a more traditional approach whereby greater emphasis is given to the underlying data which must always be preserved. Third normal form data structures place emphasis on the data management part of the Data Warehouse, and trade this off against the information access aspects.

Figure 6–35 Centralized Data Warehouse



In contrast to the conforming Data Mart architecture, a centralized architecture offers a single point of historical information management. This architecture attempts to deliver constant, consistent, and reliable cross-functional historical data across the entire organization, regardless of how it may be used by information consumers.

Like the conforming Data Mart architecture approach it can be implemented incrementally, one subject area or business process at a time. However, rather than creating separate Data Marts for each subject area, all subject areas are combined into a single Enterprise Data Warehouse (EDW). The EDW offers a single integrated data source - a "single source of truth". Users can access the EDW for both a subject area query or an overall enterprise view.

This architecture preserves a detailed record of each atomic record with rich encoding of attributes and all relationships between data elements. But, users typically require a solid understanding of the data model in order to reliably navigate the more elaborate structure.

Data are acquired from operational systems and moved into the Staging Area. If available, data from master data stores and operational data stores are also acquired and moved into the Staging Area. This Staging Area is usually in 3rd normal form and optionally can be permanent rather than temporary. By employing a permanent Staging Area it may be possible to reload the EDW without having to go back to the original data sources, which may have already purged some historical records. This might require the Staging Area to store data that is not currently required but may be needed in the future.

The Staging Area is utilized to capture, integrate, consolidate, and transform data, perform referential integrity checks, and provide cleansed quality data. The Staging Area is not intended to be queried by end users. Once the data has been processed in the Staging Area, it can be moved into the EDW within the Foundation Area.

Since both environments are structured in 3rd normal form, the movement requires little data conversion. Often the data ingestion involves an UPSERT process. An UPSERT process is updating the row in the EDW if it already exists, or inserting it if it does not already exist.

A common practice is to extend this centralized architecture to include Data Marts and form a hub and spoke architecture. Subsets of the data from the EDW are copied to the Data Marts. These Data Marts are called dependent Data Marts because they are dependent upon the EDW for their data and semantics. These Data Marts may contain summary level data, depending upon the business requirements. Refer to Section "Hub and Spoke" for details on the hub and spoke architecture.

Benefits

By utilizing a permanent Staging Area containing data in 3rd normal form, the data acquisition process is more flexible and easier to implement compared to the temporary Staging Area used in other approaches.

The EDW is the single source of truth. If dependent Data Marts are employed, the queries will not conflict with the EDW as these Data Marts have been populated with data from the EDW. In addition, the EDW can support other initiatives such as operational reporting and data mining.

Even though the data model for the EDW takes more effort, the upfront analysis and design produces a more complete model. In addition, once the design effort of the EDW is complete, the effort to generate more Data Marts is lessened.

Challenges

While an EDW can be deployed incrementally, it requires much more up front analysis to design a large centralized EDW to support a centralized architecture. Especially for smaller or more dynamic organizations that may lack patience or executive support for such an undertaking.

Of course, building an EDW in 3rd normal form does not guarantee it is immune to business changes. For example, a hierarchy might be simply represented with one table in the Data Warehouse for each hierarchy level, but what happens if the business decides to change the number of levels in the hierarchy, or for some levels to be skipped for a certain department or product? Both these changes would require a change to the physical data model and this inevitably also makes retaining a 'what was' reporting perspective problematic. In this case, something simple like a 'bill of materials' structure with a level table would avoid these structural changes.

The permanent Staging Area can grow very large very quickly, which in turn requires much more hardware resources than adopting a temporary Staging Area.

The biggest challenge of this approach lies with the data not being held in a user-friendly format. The EDW stores its data in 3rd normal form, which is not very easy for users to understand in comparison to a star dimensional structure. Although the centralized Data Warehouse is superior in terms of historical data management, it is not nearly as easy to use for analytical operations. Lacking dimensional models, the Data Warehouse is not easy to navigate - either for users or for the tools they tend to use. Therefore, database and software development expertise is often required in order to produce even the most basic analysis, dashboards, and reports.

6.3.4.3.3 Hub and Spoke

Business needs, as well as database technologies and tools, have moved on considerably since data warehousing became popular. No longer isolated, Data Warehouses are increasingly seen as a fundamental element of every business process, and part of the platform that enables business execution by providing insight into operational systems.

The hub and spoke architecture can be defined in various forms, depending on which areas are utilized, the format of the data within the areas, type of access given to each area, whether permanent and temporary storage is utilized, and the frequency of

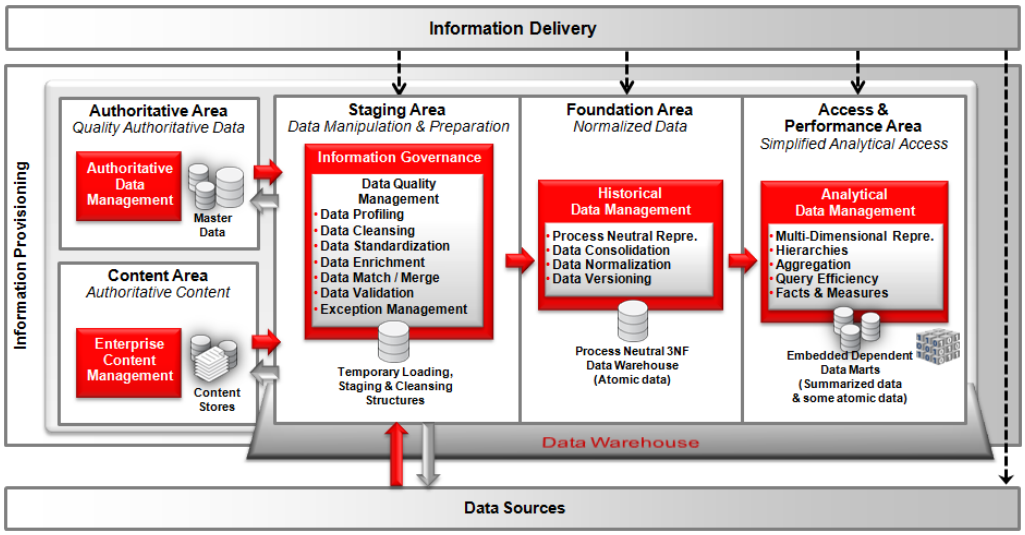
movement of data from the source and between the areas. Example 6–36 is just one example of a hub and spoke architecture which utilizes aspects of other Data Warehouse architectures and therefore can also be referred to as a hybrid approach.

The intent of the hybrid hub and spoke approach is to combine the historical data management benefits of the centralized approach with the access and navigation benefits of the conformed Data Marts approach. Oracle products are flexible and powerful enough to support all of the Data Warehouse patterns discussed in this document. But, for over a decade, Oracle's customers have seen many benefits and successful implementations utilizing the hybrid hub and spoke architecture. Therefore, it forms a key part of Oracle's information management architecture.

The key concept of this hybrid approach is the separation of concerns between historical data management and analytical operations. In short, it leverages a centralized Data Warehouse to manage historical data in a business process-neutral data model, and exposes data for analytical operations via dimensional models and cubes. The Foundation Area maintains history, while the Access and Performance Area provides dimensional models and cubes that change over time to reflect current business processes and hierarchies.

To see the following hub and spoke architecture in context, refer to the *ORA Business Analytics Infrastructure* document.

Figure 6–36 Hub and Spoke Data Warehouse



Data are acquired from various data sources and are assumed to be un-cleansed, unfiltered, and potentially incomplete. Data can originate from both internal and external sources. They are generally structured data, but may include unstructured/semi-structured content such as images, documents, sound files, spatial data, etc. Acquired data initially lands in the Staging Area which contains both temporary and permanent structures. (See Section "Information Acquistion" for more details on data acquisition techniques.)

Acquired data may also include master data. The Authoritative Area contains master data that represents many of the major dimensions supported in the Data Warehouse. While the Authoritative Area maintains the current master, the Data Warehouse can still be the recipient of the mastered data and record changes to the mastered data over time. This will lead to better quality dimension data within the Data Warehouse, which will improve reporting. It is important to note that if a master data management

solution exists, that the Data Warehouse is the recipient of the master data and not the controller if it.

In addition to master data, the Data Warehouse is an ideal place to "associate" unstructured data with structured data in order to observe the relationships between them. Managed content can be discovered and accessed via the Content Area. The Content Area's versioning capabilities allow the content to stay within a content store within the Content Area rather than making a copy of the content and storing it in the Data Warehouse.

As shown in Example 6–36 via grey arrows, the relationship between the areas can be bi-directional by 'Operationalizing' the Data Warehouse. Examples include:

- Populating master data attributes derived by analytical processing in the Data Warehouse and by the assorted analytical tools. This information becomes immediately available to the connected applications and business processes.
- Utilizing data from the Data Warehouse such as quality business entities as the basis for defining the taxonomy to categorize content.
- Utilizing quality data from the Data Warehouse to perform data remediation of the source systems.
- Utilizing the results of analysis, e.g. statistical analysis and data mining, as inputs to operational processes. Such results might be used to recommend purchases, detect fraud, or to adjust pricing.

Once the data has been processed in the Staging Area it can be moved into the Foundation Area. The Foundation Area represents the heart of the Data Warehouse and is the area responsible for managing the data over the long term, including versioning and history. Data in the Foundation Area is kept for as long as possible. This requires some form of information lifecycle management with different levels of compression or different tiers of storage to minimize the cost of ownership. This is better than archiving data aggressively as it means more information is available to be queried.

The Foundation Area is in 3rd normal form, and stores atomic level detail designed using relational schema patterns that easily allow for changes to business objects and hierarchies. This allows it to evolve in support of business changes. It is advantageous to leverage an enterprise information model sourced from one of the industry bodies or an EDW model from database vendors such as Oracle or a specialist model vendor. Some element of adaptation of the logical model to meet local needs is typically required, and any process-oriented representations included in the model must be removed before it can be transformed into a final physical model.

As previously discussed, the 3rd normal form model approach used to achieve the data management goals of the Foundation Area is not helpful when it comes to providing users access to the data, as it is more difficult to navigate.

The Access and Performance Area adds the access components to the architecture. It enables responsive queries via several mechanisms such as indices, materialized views, columnar data storage, and in-memory data management.

The majority of queries will of course be against the Access and Performance Area, as its responsibility is to simplify user access to the data. But an architect may decide that it makes sense that, in a controlled manner, all areas are available to users. For example, aspects of the Staging Area, such as data quality information, can be made available to end users. Data quality issues should be visible to all users since data quality is not only an issue for IT.

As mentioned earlier, to see this architecture pattern in context, refer to the *ORA Business Analytics Infrastructure* document. As this hub and spoke pattern is a hybrid of the centralized and conformed Data Mart patterns, it shares many of the same benefits and challenges.

Benefits

It could be stated that this is the ideal Data Warehouse architecture as it unites the operational and analytical sides of the business. A true single view is possible. The data driving the reporting and decision making is actually the same data that is driving the operational applications. Derived information on the analytical side of the business is made available to the real-time processes using the operational applications and business process orchestration tools that run the business. While at the same time, business changes have little or no impact on historical data, yet business users can operate on the most efficient data representations.

Invariably, the data held in source systems will be in a relational format, which aligns naturally with the normalized and relational format of the Foundation Area. This makes data acquisition and transformation quicker and easier than if the data had to transform immediately into a dimensional model. The transformation from a relational model to a dimensional model can be executed at a later stage without affecting the data acquisition process.

As previously discussed, the 3rd normal form model approach used to achieve the data management goals of the Foundation Area is not helpful when it comes to providing users access to the data as it is more difficult to navigate. While star schema and dimensional models are advantageous for easier consumption by end users, they are not conducive to change. But, by adhering to the principle of 'separation of concerns', the Foundation Area can support the Access and Performance Area, where star schemas and dimensional models can be reconstructed and re-populated using historical data in a timely fashion based on new or changed business requirements.

Most critical though is the realization that the tools used to access the Data Warehouse will change over time. Architects have very little control over which tools are adopted by a particular business unit. As new tools may impose different requirements in terms of the way that analytical data are structured, it is critically important that it is possible to create analytical models (or re-create them) from the underlying pool of data.

Even though we have described the areas as separate areas of responsibilities, this does not mean that they have to be physically separated. This means that the Data Marts can, in effect, be embedded dependent Data Marts. Several advantages accrue through embedding these structures rather than pushing the data out into a separate downstream Data Marts. There is no need to buy a completely separate hardware platform on which to run the Data Marts. In addition, data does not need to be offloaded, transported, re-loaded and aggregated onto a second platform. This reduces latency and can improve accuracy, as well since the data are less stale. The use of a single DBMS system helps to promote consistency by eliminating the need to manually enforce conformity across multiple Data Marts. Lastly, it also helps to reduce complexity by eliminating many of the integration concerns that can be attributed to physically separated data.

For some infrastructure, this logical movement of data can be achieved via a metadata change rather than more costly wholesale data ingestion. The majority of the Access and Performance Area is made up of objects that will refresh automatically. For instance, this is true for views, materialized views and cube organized materialized views. In these two latter cases, their definition will also define how they are refreshed, either when the data becomes stale or when a user queries the view, thus deferring the aggregation until sometime later.

Challenges

It is important that any hybrid architecture is accompanied by a supporting method that can deliver business results in an incremental fashion. This method should focus on developing a Foundational Area which does not impact the delivery and usability of the Access and Performance Area. Therefore, it usually takes an organization that has a good level of maturity with data modeling, integration processes, role definitions, and business and IT information infrastructure and tools.

The accuracy of information that is provisioned for the sake of historical management must not be compromised by the changes that occur within the organization from day to day or year to year. Likewise, usability of analytical information should not suffer for the sake of historical integrity.

6.3.4.3.4 Federated and Distributed

Although it is best to always aim for the minimum number of Data Warehouses, (ideally one), there are exceptions to this rule. For instance, large and sufficiently siloed organizations may have a number of Data Warehouses. Likewise, merger and acquisition activity may lead to multiple Data Warehouses. But ultimately if the main concern is one single source of the truth, a single enterprise Data Warehouse is what should be aimed for.

Utilizing either a federated or distributed approach to Data Warehouses may prove beneficial to migrating towards a single Data Warehouse, or as an approach to integrate and/or access information between the Data Warehouses.

Each Data Warehouse can be seen as simply data sources and accessed using federated queries (See Section "Federation"). This approach does have its advantages as it is quick to implement and leaves the existing data in the warehouses. But complex queries and survivorship rules can make a federated approach challenging. In addition, a federated approach can suffer from poor performance and therefore a federated approach should only be used when it makes sense; where performance is acceptable and the information consumer can't wait for data to be consolidated into a single Data Warehouse.

The use of Information Services allow an organization the flexibility to initially access data via a federated approach, and over time as data are consolidated, to later access data from a Data Warehouse.

A distributed approach utilizes architecture patterns similar to those highlighted earlier in Section "Master Data Management Patterns" (e.g. virtual, centralized) utilizing key references, unique identifiers, survivorship rules, and global systems of record to address the challenges with integrating and accessing data across Data Warehouses.

6.3.4.4 Logical Data Warehouse

As previously stated, a traditional Data Warehouse can be seen as a primary storehouse of primarily historical structured data that has been cleansed, transformed, and consolidated for an organization. This provides business users with a time-based, integrated view of cross-functional data that supports decision making by enabling reporting and analytics.

A Data Warehouse initiative should be seen as a program and not a project. This infers that developing a Data Warehouse is never complete. New data sources are identified, existing data sources disappear or become less relevant, and new and unknown questions are being asked by the business. Therefore Data Warehouses and the associated information management architecture need to constantly evolve to cater for these known and currently unknown requirements, such as Big Data.

It would seem that the simplest approach accommodate Big Data within a traditional Data Warehouse is by simply seeing Big Data as "another" data source. But the traditional Data Warehouse would invariably encounter processing challenges pertaining to volume, velocity and variety.

6.3.4.4.1 Siloed Big Data Architecture

As in the case with many new technology strategies, Big Data processing and analytics (aka Big Data Warehouse) brings new capabilities and benefits to an organization even when deployed in a silo. In the case of a Big Data Warehouse, it is the ability to handle data sets whose size, structure, and acquisition speed are beyond the capability of conventionally used (capture, ingest, manage, process, and analyze) software tools..

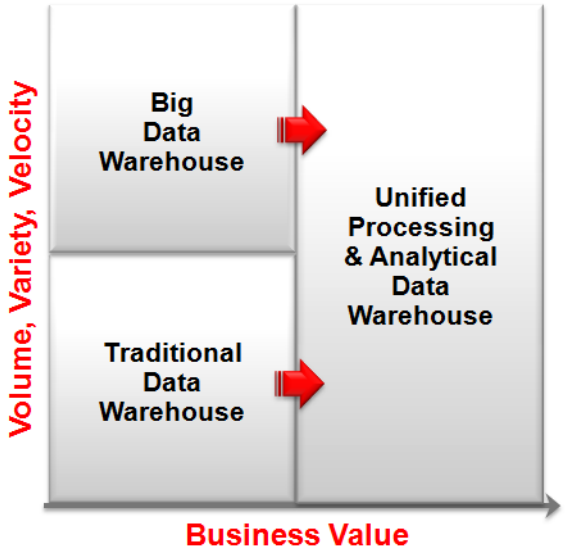
While processing Big Data in isolation brings some benefit, only companies that fully integrate these new types of information with traditional corporate data, and fit the insight they glean into their existing business processes and operations, will truly see the full value of Big Data.

6.3.4.4.2 Unified Processing and Analytical Architecture

As discussed in Section 6.3.2, Big Data is changing the business analytical landscape, and in turn, Data Warehouses. "One-size-fits-all" data processing is impractical to address all analytical requirements. Therefore, it is important to consider how your information management and Data Warehouse architecture will not only coalesce data from new and traditional data sources, but also unify the new and traditional business analytical infrastructures to support descriptive, predictive, prescriptive and exploratory analysis.

The architecture to support Big Data Processing and analytics must not be seen as a siloed initiative or a bolt-on for a traditional Data Warehouse. As illustrated in figure x below, organizations must transition to a unified data processing and analytical architecture to fully coalesce "all" corporate data that can be analyzed so that companies gain a true 360-degree view of their organization.

Figure 6–37 Unified Processing & Analytical Data Warehouse Positioning



A term that has grown in use recently is the Logical Data Warehouse (LDW). Gartner defines LDW, as a new data management architecture for analytics which combines the strengths of traditional repository warehouses with alternative data management

and access strategies.[3] While still early in its maturation, this definition satisfies the high-level goals of a unified processing and analytic Data Warehouse. For the remainder of this section the term Logical Data Warehouse will be used.

A logical Data Warehouse does not emphasize a single repository or persistence model, but a variety of data repositories and processing engines that are fit for purpose; whereby business needs, analytic workloads, costs and processing engines are matched accordingly.

The following table highlights the characteristics of coalescing the data and analytical capabilities into one logical information delivery platform.

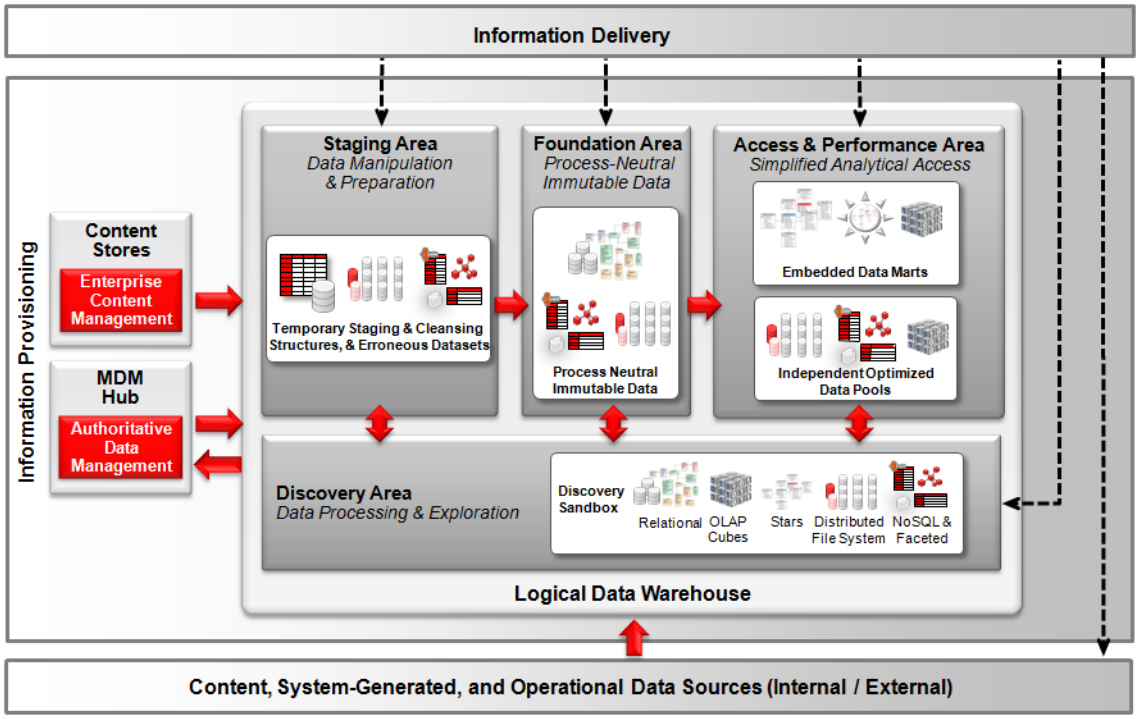
Table 6–10 Logical Data Warehouse Comparasion

	Traditional Data Warehouse	Big Data Warehouse	Logical Data Warehouse
Ingestion	Primarily ETL/ELT/CDC	Primarily ingestion of raw data via ETL (MapReduce) & event capture	All ingestion mechanisms & access to raw data
Data	Integrated & Consolidated	Raw, Aggregated	Integrated, aggregated, consolidated, & virtualized
Structure	Primarily Structured Data	Strong focus on unstructured & semi-structured data	Multi-structured
Data Quality	High (Process & Cleanse)	Low (Process & Cleanse - if & when required)	All levels of quality data available from uncleansed raw datra to highly cleansed data
Data Modeling	Upfront schema modeling leading to process neutral & normalized data	Parse schema on demand & "model as you go" for exploratory analytics.	Model as appropriate as requirements demand
Processing Engines	Online analytical processing	Parallel Batch Processing Engine, Event Processing Engine	All available processing engines.

Consolidating the areas of responsibilities and architecture capabilities covered earlier in this document (i.e. Section "Big Data Processing", Section "Data Warehouse"), the following architecture diagrams can be defined. The red arrows represent the use of one or more ingestion capabilities described in Section 6.2.1.

[3] Gartner Blog Post - Mark Beyer - Nov 2011

Figure 6–38 Logical Data Warehouse - Areas of Responsibility



Data can be ingested from data sources within the Information Provisioning layer or from a number of internal/external sources including system generated data from devices, content from file systems, and operational systems. These data are either ingested into a Staging Area where it is prepared for historical online storage, or into a Discovery Area where it can be processed and analyzed.

The Staging Area acts as a landing zone as it enters the LDW for all forms of data, whether they are managed or not, (e.g. content from a file system vs. content from a content management system). The data is cleansed according to requirements (if at all).

Data, once properly staged, flows from the Staging Area to the Foundation Area. The timing is dependent upon when staged data are ready (e.g. processed, filtered, complete, cleansed, and compliant). Requirements and data formats dictate what level of processing is required. Not all data will need to be cleansed to the same level of quality. When moving data between areas, it need not be an all-or-nothing flow (e.g. customer churn data could be updated hourly, while sales data are updated daily).

The Foundation Area acts as the historical system of record. The intent is to maintain data in a form that is both efficient (for managing large volumes of data), and versatile (such that schema changes are extremely rare). For structured data, a normalized business-neutral model is preferred. This supports storage at the finest level of granularity for as long as necessary without a need to make changes when business processes or business hierarchies change. For less structured data where normalization is not feasible, such as log data or sensor data, historical storage may simply be a collection point of historical record without versioning or relational reference capabilities.

As previously highlighted, the rightmost area is the Access and Performance Area. Here, data are managed in a way that best serves the consumers. For structured data, this typically involves a dimensional storage model that may include aggregations and/or cubes. Dimensional models are easiest to navigate. Unlike the Foundation

Area model, dimensional models directly reflect the current business hierarchies. This makes them prone to changes over time as the business evolves. For less structured data, this area may be implemented as a subset of historical data with or without additional metadata, indexing, or relationships added that enhance accessibility and query performance.

The Discovery Area provisions and manages a number of investigative data stores (aka Sandboxes) to enable analytics. These sandboxes tend to be provisioned, altered, removed, and owned by data analysts/scientists and typically have a limited lifespan. The Discovery Area ingests data from various downstream and upstream sources (e.g. all areas and raw data sources) in support of the business problem at hand. This data is processed and analyzed using any and all tools at the disposal of the Data Scientist, and therefore can often be resource and processing intensive.

The output gleaned from the analytical tasks performed with in the Discovery Area may be ingested into the other areas of the architecture. Generally, this output will enter the Staging Area where it can be managed and propagated along with other data that is destined for historical and/or analytical storage. Output may also be ingested directly to the Foundation Area if staging is not necessary. This will likely be the case for unstructured data, e.g. Big Data that is simply being collected for historical record without versioning and normalization. Data may also be ingested directly to the Access and Performance area, provided it does not need to be included in the historical record. For example, temporary information such as marketing segments or quartiles may be periodically calculated and added to a cube for market analysis.

Figure 6–39 Logical Data Warehouse Components

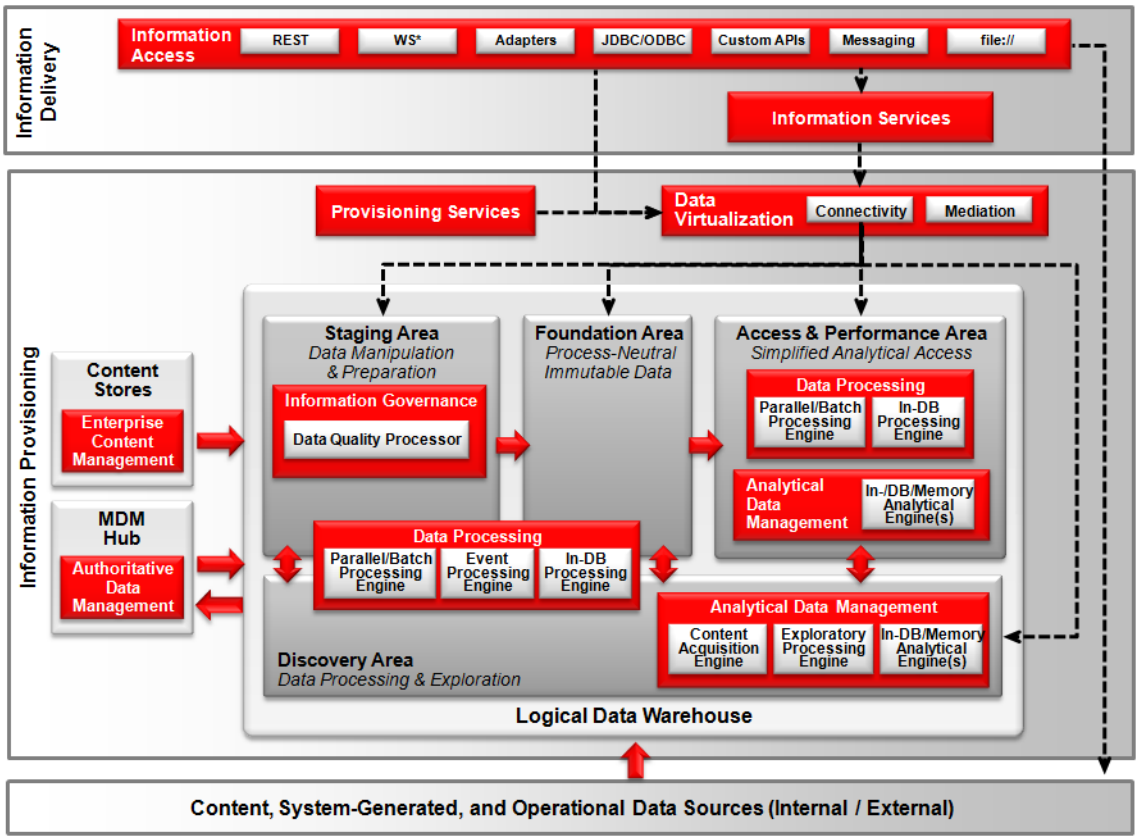


Figure 6–39 above highlights the interaction between key information management components utilized as part of a LDW. As previously highlighted, data can be

collected from a number of internal/external sources including system-generated data from devices, content from file systems, and operational systems. These data are either ingested into a Staging Area where it is prepared for historical online storage, or into a discovery area where it can be processed and analyzed in either a (micro) batch or trickle feed manner.

The LDW utilizes a number of components. As can be seen from the diagram above, the components are an accumulation of the components utilized and previously described in the Big Data Processing section and Hub and Spoke Data Warehouse section.

The LDW utilizes In-DB Processing and In-DB Analytical components. They enable various forms of data processing, as well as analytical processing, to be performed on data without extracting it from the database. This greatly enhances performance and reduces overhead on the network by applying the processing steps where the data physically reside. Examples include in-DB transformation (i.e. the 'T' in ELT), statistical analysis, data mining, and text mining.

To meet extreme query response requirements, the LDW utilizes in-memory analytical components that enable high speed query processing, results caching, and OLAP.

The LDW also makes use of processing components that exist outside of the database. Examples of this include Parallel Processing, such as MapReduce jobs that are executed on distributed data stores, and event Processing Engines that enable real-time data processing.

To support exploratory analytics, the LDW utilizes a Content Acquisition Engine and an Exploratory Processing Engine. The Content Acquisition Engine provisions crawled data from multiple data sources such as file systems, content management systems, and web servers. This crawled data is intelligently tagged and made available for the purposes of Exploratory Analytics. To support intuitive exploration and analysis of information, an Exploratory Processing Engine provides unprecedented flexibility in combining diverse and changing data, as well as extreme performance in analyzing data, by utilizing a hybrid, search/analytical database.

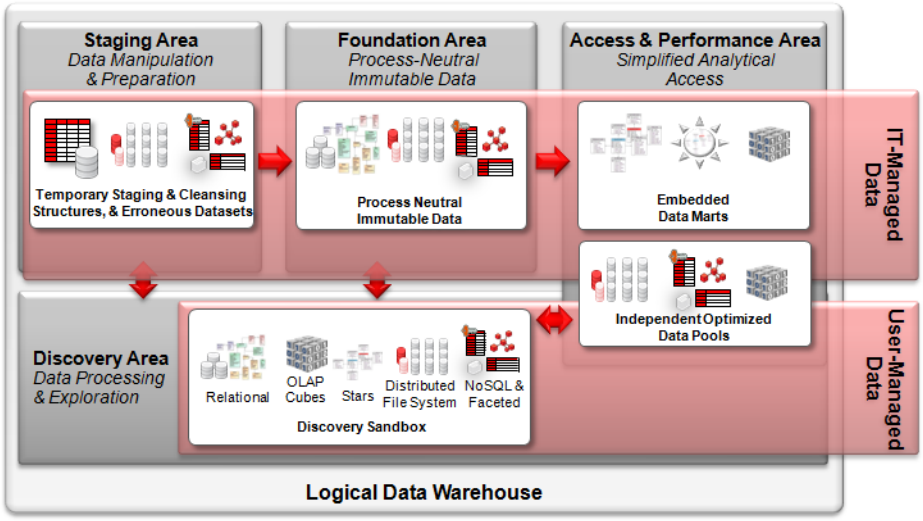
Lastly, a data quality processor is utilized within the staging area as a powerful rules-based engine that executes a number of processes to cleanse, standardize, enrich, match, and de-duplicate any type of data, including names and addresses. Refer to Section "Data Quality Processor" for more details.

Refer to the *ITSO Business Analytic*s documents for further information regarding a Logical Data Warehouse.

6.3.4.4.3 Versatility versus Consistency

A key objective of Data Warehouses is to create an infrastructure where consistency can be established and maintained. In previous sections, this has been addressed by the definition of Staging and Foundation Areas where the formalization and operationalization of cleansing and historical data management takes place. Generally, this benefits most forms of analysis by ensuring that complete and accurate data are always available, and properly versioned, spanning across all business processes. A drawback of achieving this consistency is that it may impede progress. New data elements, or changes to data models, or the ingesting of new data sources, tend to take a rather long time to be fulfilled. Once this new data has been ingested it may yet have to go through a cleansing process, which delays its availability even more. An environment in which this level of consistency and operationalization is achieved is invariably managed by the IT Department (See Figure 6–40).

Figure 6–40 IT Managed and User Managed Data



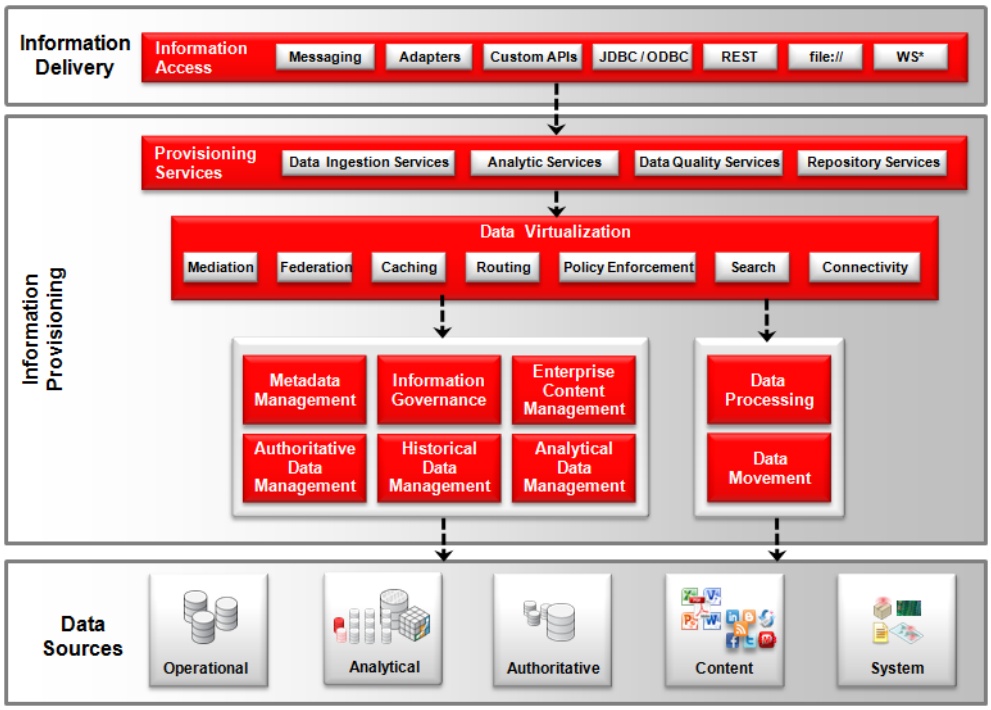
While most forms of analysis benefit greatly from consistent, well-structured, and well-governed data, other forms of analysis demand a more versatile environment. This versatile environment allows and enables users to provision and modify data on the fly, and is less regimented in terms of governance. As the definition suggests, these environments are primarily managed by the end user.

Refer to the *ITSO ORA Business Analytics Infrastructure* document for more details on IT Managed and User Managed environments.

6.3.5 Information Provisioning as a Service

Information Provisioning as a Service is a service-oriented approach to provisioning information in a consistent and secure manner (See Figure 6–41).

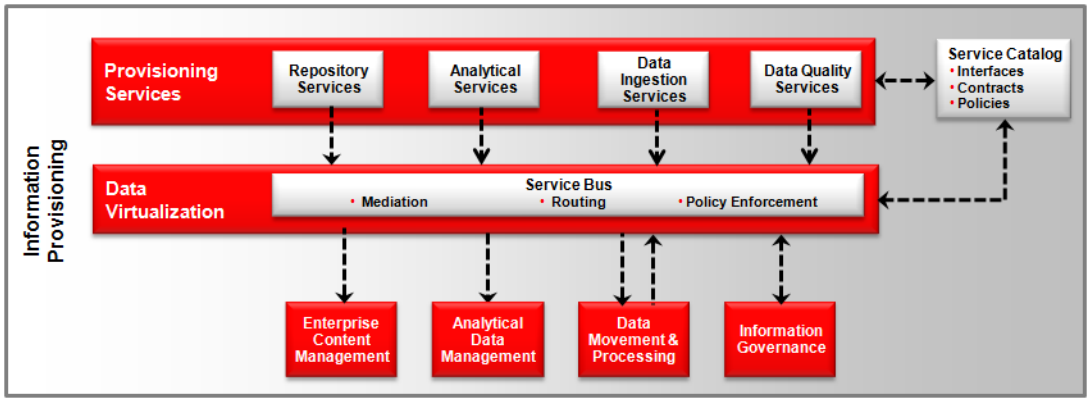
Figure 6–41 Information Provisioning as a Service



The implementation of Provisioning Services enables the logical abstraction of underlying architectural concerns and presents the provisioning of both operational and analytical information via a clearly defined, pre-integrated, and simplified interface.

Provisioning Services are services that act solely on the capabilities within the Information Provisioning Layer - unlike Information Services which focus primarily on the consumption of information. In effect, Provisioning Services are services that operate on information management that expose highly optimized engines for the provisioning and distribution of data. Therefore the underlying architectural concerns of the Information Provisioning Layer have been abstracted and presented in a clearly defined interface.

Figure 6–42 Provisioning Services



Provisioning Services represent provisioning functionality that has been pre-configured for a specific purpose. The request and response of these services is formalized in a contract and/or interface specification. They offer the benefits of being

pre-engineered (tested, documented, and governed), to standardize results and maximize the efficiency and performance of common, known provisioning activities. Here are some examples of Provisioning Services:

- **Data Ingestion Services** - Data Ingestion Services provide data ingestion and transformation capabilities. These services can be utilized by BPEL/ESB process and/or SOA Services to initiate the ingestion and transformation of data.
- **Data Quality Services** - Data Quality Services use algorithms and pre-defined business rules to clean up, reformat, and de-duplicate business data. These services can be utilized in a variety of ways. Some examples include:
  - Using Data Quality Services to apply cleansing and standardization capabilities to messages for real-time and preventive usage. This enables a proactive approach to make sure that operational applications cleanse data before there is a problem in downstream usage.
  - Using Data Quality Services in conjunction with other Provisioning Services such as Data Movement Services to cleanse data acquired from operational systems as it is consolidated into a historical data store such as a Data Warehouse.
  - Using Data Quality Services to cleanse data within a data source (e.g. cleaning up a legacy database).
- **Repository Services** - Repository Services deliver functionality offered by repository stores such as content stores. Examples include workflow services and conversion services.
- **Analytical Services** - Analytic Services provide functionality to enable analytic provisioning. Examples include the creation, copying and deletion of dimensions.

These Provisioning Services are cataloged in a way that supports discovery and promotes reuse. The interface specifications, usage policies, access policies, etc., are maintained in a service catalog.

All Provisioning Service requests are directed through the Data Virtualization layer. This layer provides several capabilities, most of which are optional and will not be required for all queries. However, abstraction is highly recommended as a means to avoid tight coupling between provisioning service consumers and the underlying provisioning engine. The Data Virtualization Layer can route the request to one of several locations depending on the Provisioning Service request.

It is also recommended that calls between engines are routed through the virtualization layer to de-couple the engines giving the flexibility to upgrade or change the engine in the future.

7

Product Mapping View

This chapter introduces several products from Oracle and describes how they map onto the example scenarios described in the logical view. The intention is to position products based on their primary strengths, and to demonstrate how they can be combined to satisfy higher level business and IT objectives.

Oracle has a comprehensive list of products that support information management. This document covers the core products. Details on additional products that are relevant in the information management space can be found in the following documents:

- For the mapping of enterprise performance management and business intelligence products, please consult the *ORA Business Analytics Infrastructure* document.
- For the mapping of security products, please consult the *ORA Security* document.
- For the mapping of management products, please consult the *ORA Management and Monitoring* document.
- For the mapping of SOA products, please consult the *ORA SOA Infrastructure* document.
- For the mapping of EDA products, please consult the *ORA EDA Infrastructure* document.
- For the mapping of engineered systems, please consult the *ORA Engineered Systems* document

The logical view of the architecture has been designed in a modular fashion, with a degree of segregation of duties, and a focus on open standards. Therefore although this view only includes products from Oracle, it can be customized to position a mix of products from different vendors.

> **Note:** Product names and features will evolve over time. This section presents information that is current as of the writing of this document. Please consult Oracle's online product documentation for details on product capabilities and features for specific product versions.

7.1 Products

7.1.1 Product Classification

The following tables classify the Oracle products and specific options mentioned earlier in this chapter by the various areas of the ORA Information Management

architecture. These tables are then followed by an alphabetical description of the aforementioned products.

7.1.1.1 Information Delivery

Table 7–1 Information Delivery Product Mapping

Area	Products
Information Consumers & Producers	■ Oracle has a comprehensive range of products that can be utilized as information consumers and producers. NOTE - Consumers can be producers and vice versa. ■ Consult *ORA Business Analytics Infrastructure* document for BA examples
Information Access & Services	■ Oracle has a comprehensive range of products that can be utilized to develop and access services 9e.h. Oracle SOA Suite, Oracle Application Grid). In addition a number of products listed in this table also include pre-built Information services.

7.1.1.2 Information Provisioning

Table 7–2 Information Provisioning Product Mapping

Area	Products
Provisioning Services	■ Oracle has a comprehenisve range of products that can beutilized to develop services (e.g. Oracle SOA Suite, Application Grid). In addition a number of the products listed in this table also include pre-built provisioning services.
Data Virtualization	■ Oracle BI Server ■ Oracle Coherence ■ Oracle Data Service Integrator ■ Oracle Database Gateways ■ Oracle Direct Connector for Hadoop Distributed File System ■ Oracle Secure Enterprise Search ■ Oracle Service Bus ■ Oracle Service Registry ■ Oracle Webcenter Content
Information Governance	■ Oracle Enterprise Data Quality ■ (Product) Match & Merge ■ (Product) parsing & Standardization ■ Profile & Audit ■ Oracle Webcenter Content
Enterprise Content Management	■ Oracle Secure Enterprise Search ■ Oracle Webcenter Content

Table 7–2 (Cont.) Information Provisioning Product Mapping

Area	Products
Authoritative Data Management	■ Oracle MDM Hubs ■ Oracle Customer Hub ■ Oracle Supplier Hub ■ Oracle Product Hub ■ Oracle Site Hub ■ Oracle Higher Education Constituent Hub ■ Oracle Data Relationship Management (Financial, Analytical) ■ Oracle Application Integration Architecture
Historical & Analytical Data Management	■ Oracle Data Models (Airline, Communications, Retail) ■ Oracle Database (EE) ■ Advanced Analytics ■ Advanced Compression ■ Partitioning ■ OLAP ■ Spatial ■ Oracle Essbase ■ Oracle Exadata ■ Database Machine ■ Intelligent Warehouse Solutions
Data Movement	■ Oracle Active Data Guard ■ Oracle Data Integrator (EE) ■ Oracle Data Integration Application Adaptor for Hadoop ■ Oracle Loader for Hadoop ■ Oracle Database (EE) ■ Advanced Compression ■ Partitioning ■ Oracle Golden Gate ■ Cloudera Flume ■ Cloudera Oozie ■ Cloudera Sqoop

Table 7–2 (Cont.) Information Provisioning Product Mapping

Area	Products
Data Processing	■ Cloudera MapReduce ■ Oracle Event Processing ■ Hadoop/NoSQL Cartridge ■ Spatial Cartridge ■ Oracle Big Data Appliance ■ Oracle Coherence ■ Oracle Endeca ■ Server ■ Content Management System Connectors ■ Text Enrichment (with Sentiment Analysis)

7.1.1.3 Data Sources, Modeling, Security & Management

Table 7–3 Data Sources, Modeling, Security & Management Product Mapping

Area	Products
Data Stores	■ Oracle has a comprehensive range of products that can beutilized as data stores including areas such as (IMDB) databases, web services, flat files, repositories, packaged / bespoke horizontal applications and vertical applications. ■ Oracle NoSQL Database ■ Cloudera HDFS ■ Cloudera HBase
Information Modeling & Metadata Management	■ Oracle Data Models (Airline, Communicationsm Retail) ■ Oracle Enterprise Repository ■ Oracle SQL Developer Data Modeler ■ Each product listed in the above tables contain embedded metadata capabilities
Security	■ Consult *ORA Security* document
Management & Monitoring	■ Cloudera Hue ■ Consult *ORA Management & Monitoring* document

7.1.2 Product List

- **Oracle Application Integration Architecture (OAIA)** - Oracle Application Integration Architecture (AIA) Process Integration Packs (PIPs) are prebuilt integrations across best of breed Oracle applications, like Siebel CRM, Oracle E-Business Suite, Agile PLM, Oracle Communications Billing and Revenue Management (Portal), and SAP, enabling customers to rapidly deploy fully orchestrated business processes. AIA PIPs make it easier for customers to implement core business processes and spend less time maintaining/upgrading integrations.
- **Oracle Big Data Appliance (BDA)** - Oracle Big Data Appliance brings Big Data solutions to mainstream enterprises. Built using industry-standard hardware from

Sun and Cloudera's Distribution including Apache Hadoop, Big Data Appliance is designed and optimized for big data workloads. By integrating the key components of a Big Data platform into a single product, Oracle Big Data Appliance delivers an affordable, scalable and fully supported Big Data infrastructure without the risks of a custom built solution. Oracle Big Data Appliance integrates tightly with Oracle Exadata and Oracle Database using Oracle Big Data Connectors, and seamlessly enables analysis of all data in the enterprise - structured and unstructured.

– **Cloudera Hadoop Distribution (CDH)** - Oracle Big Data Appliance includes Cloudera's distribution of Apache Hadoop and a number of other packages for the Hadoop ecosystem. CDH is thoroughly tested and certified against Oracle Big Data Appliance. CDH contains the follows packages:-

  * **Apache Hadoop**

    HDFS - The Hadoop Distributed File System (HDFS) is the primary storage system used by Hadoop applications. HDFS creates multiple replicas of data blocks and distributes them on compute nodes throughout a cluster to enable reliability, and extremely rapid computations.

    MapReduce - Above HDFS sits a framework called Apache MapReduce for parallel processing of large data sets across a large number of nodes. Computational processing can occur on data stored either in a file system (unstructured) or in a database (structured). MapReduce can take advantage of locality of data, processing data on or near the storage assets to decrease transmission of data.

  * **Apache Flume** - Flume is a distributed, reliable service for efficiently collecting, aggregating, and moving large amounts of log data from many different sources to a centralized data store. It has a simple and flexible architecture based on streaming data flows. It is robust and fault tolerant with tunable reliability mechanisms and many failover and recovery mechanisms.

  * **Apache HBase** - HBase is a distributed NoSQL column-oriented store built on top of HDFS providing the capability to perform consistent real-time random read/write access to very large data sets.

  * **Apache Hive** - Hive is a system that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. At the same time, this language also allows traditional MapReduce programmers to plug in their custom mappers and reducers when it is inconvenient or inefficient to express logic in HiveQL.

  * **Hue** - Hue is a browser-based environment that enables users to interact with a Hadoop cluster. Hue includes several easy to use applications that assist users in working with Hadoop MapReduce jobs, Hive queries, and user accounts. The Hue applications run in a Web browser

  * **Apache Mahout** - Mahout offers a number of core machine learning algorithms such as clustering, categorization, and collaborative filtering that are implemented on top of Apache Hadoop using MapReduce.

  * **Apache Oozie** - Oozie is a scalable, reliable, and extensible workflow/coordination system to manage several types of Apache Hadoop jobs (e.g. Java MapReduce, Streaming MapReduce, Pig, Distcp,

etc.). Oozie enables execution of tasks via multiple action options e.g. Java action, MapReduce action, Pig action, Fs action, and so on.

* **Apache Pig** - Pig is a platform for analyzing large data sets that consists of a high-level language called Pig Latin coupled with an infrastructure that consists of a compiler that produces sequences of MapReduce programs. Pig programs are amenable to substantial parallelization, which in turn enables them to handle very large data sets.

* **Apache Sqoop** - Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases. Sqoop can be utilized to import data from external structured datastores into Hadoop Distributed File System or related systems like Hive and HBase. In addition, Sqoop can be used to extract data from Hadoop and export it to external structured datastores such as relational databases and enterprise Data Warehouses.

* **Apache ZooKeeper** - ZooKeeper is a high-performance coordination service for distributed applications. It exposes common services, such as naming, configuration management, synchronization, and group services. ZooKeeper offers a simple interface so that it can be utilized to enable distributed processes to coordinate with each other through a shared hierarchal namespace which is organized similarly to a standard file system.

- **Oracle Business Rules** - Oracle Business Rules is a high performance lightweight business rules product that addresses the requirements for agility, business control, and transparency. It is part of the Fusion Middleware stack and integrates seamlessly across the entire Oracle SOA Suite and BPM Suite stack.
- **Oracle Coherence** - Oracle Coherence is the industry's leading in-memory data grid solution, enabling organizations to predictably scale mission-critical applications by providing fast and reliable access to frequently used data. As a shared infrastructure, Oracle Coherence enables real-time data analysis, in-memory grid computations, parallel transaction and event processing, and application grid computing
- **Oracle Data Integrator EE** - Oracle Data Integrator Enterprise Edition is a comprehensive data integration platform that covers all data integration requirements-from high-volume, high-performance batches, to event-driven, trickle-feed integration processes, and SOA-enabled Data Services. Oracle Data Integrator Enterprise Edition addresses multiple enterprise data integration needs. Oracle Data Integrator delivers next-generation, Extract Load and Transform (E-LT) technology that improves performance, reduces data integration costs, even across heterogeneous systems. Unlike conventional ETL tools, Oracle Data Integrator EE offers the productivity of a declarative design approach, as well as the benefits of an active integration platform for seamless batch and real-time integration. In addition, hot-pluggable Knowledge Modules provide modularity, flexibility, and extensibility. Oracle Data Integrator Enterprise Edition is optimized for Oracle Database to provide real-time data warehousing with advanced ETL/ELT and data modeling.
- **Oracle Data Integration Application Adapter for Hadoop** - Provides native Hadoop integration within Oracle Data Integrator. Based on Hive, it enhances productivity and provides a simple user interface for creating programs to load data to and from files or relational data stores.
- **Oracle Data Models**

- **Oracle Airline Data Model** is a is a standards-based, industry-specific, prebuilt data warehouse database schema with associated analytic models and dashboards. Oracle Airline Data Model provides a foundation schema that is modern, relevant, topical, and addresses the key passenger data management needs of low cost carriers and traditional airlines.

  With the Oracle Airline Data Model, airlines can more easily collect and analyze data from key areas such as reservations, sales, operations, loyalty, and finance. With sophisticated trending and data mining capabilities built into the system and focused on specific airlines industry business problems, the Oracle Airline Data Model transforms airlines industry data into insightful, actionable information.

  The Oracle Airline Data Model can be used in any application environment and is easily extendable. Combining pre-built data mining, online analytical processing (OLAP), and dimensional models, you can jump-start the design and implementation of an airline data warehouse to quickly achieve a positive ROI for your data warehousing and business intelligence project.

- **Oracle Communications Data Model** is a standards-based, industry-specific, prebuilt data warehouse database schema with associated analytic models and dashboards. The Oracle Communications Data Model provides a foundation schema that is modern, relevant, topical, and addresses the needs of multiple communications industry segments, including mobile, wireline, and cable.

  With Oracle Communications Data Model, communications service providers can more easily collect and analyze data from key areas such as customer management, marketing segmentation, product development, and network health. With sophisticated trending and data mining capabilities built into the system and focused on specific communications industry business problems. Oracle Communications Data Model has been built following the communications industry's TM Forum Information Framework (SID) standard, easing interoperability with both packaged as well as legacy OSS and BSS systems.

- **Oracle Retail Data Model** is a standards-based data model designed and optimized for Oracle data warehouses. Oracle Retail Data Model combines market-leading retail application knowledge with the power of Oracle's Data Warehouse and Business Intelligence platforms. Combining pre-built data mining, online analytical processing (OLAP), and dimensional models, Oracle Retail Data Model delivers industry-specific metrics and insights that can immediately improve your bottom line. Oracle Retail Data Model can be used in any application environment and is easily extendable.

  Oracle Retail Data Model jump-starts the design and implementation of a retail data warehouse to quickly achieve a positive ROI for your data warehousing and business intelligence project with a predictable implementation effort. Oracle Retail Data Model provides the basis for operational reporting, detailed analysis, fraud detection & loss prevention, as well as forecasting, trend & predictive analysis. With pre-built Oracle OLAP and Oracle Data Mining models geared toward specific retail business problems, Oracle Retail Data Model transforms retail data into insightful, actionable information.

- **Oracle Data Service Integrator** - Oracle Data Service Integrator provides the ability to quickly develop and manage Federated Data Services for accessing single views of disparate information. Oracle Data Service Integrator is completely standards based, declarative, and enables re-usability of Data Services. Oracle Data Service Integrator is the only data federation technology that supports the

creation of bidirectional (read and write) Data Services from multiple data sources. In addition, Oracle Data Service Integrator offers the breakthrough capability of eliminating coding by graphically modeling both simple and complex updates to heterogeneous data sources.

- **Oracle Database (EE)** - Oracle Database delivers industry leading performance, scalability, security and reliability on a choice of clustered or single-servers. It provides comprehensive features to easily manage the most demanding transaction processing, business intelligence, and content management applications. Oracle Database comes with a wide range of options to extend the world's #1 database to help grow your business and meet your users' performance, security and availability service level expectations.

  Here are just some of the database options that are applicable:

  - **Active Data Guard** - Oracle Active Data Guard, with Oracle Database Enterprise Edition, enhances quality of service by offloading resource-intensive activities from a production database to one or more synchronized standby databases. Oracle Active Data Guard enables read-only access to a physical standby database for queries, sorting, reporting, Web-based access, and so on, while continuously applying changes received from the production database. Oracle Active Data Guard also enables the use of fast incremental backups when offloading backups to a standby database and can provide additional benefits of high availability and disaster protection against planned or unplanned outages at the production site.
  - **Advanced Analytics** - Oracle Advanced Analytics extends the Oracle database into a comprehensive advanced analytics platform through two major components: Oracle R Enterprise and Oracle Data Mining. With Oracle Advanced Analytics, customers have a comprehensive platform for real-time analytics that delivers insight into key business subjects. Because the data, models and results remain in the Oracle Database, data movement is eliminated, security is maximized and information latency is minimized.
    * **Oracle R Enterprise** - Oracle R Enterprise extends the database with the R programming language's library of statistical functionality, and pushes down computations to the database. R users can use their existing R development skills and tools, and scripts can now also run transparently and scale against data stored in Oracle Database.
  - **Advanced Compression** - The Oracle Advanced Compression option to Oracle Database helps manage more data in a cost-effective manner. With data volumes, on average, tripling every two years, Oracle Advanced Compression delivers compression rates of 2-4x across all types of data and applications. Storage savings from compression will cascade throughout the data center, reducing network traffic and data backups as well. And by reading fewer blocks off disk, Oracle Advanced Compression also improves query performance.
  - **Partitioning** - A key requirement of high performance, high availability database environments, partitioning splits tables and indexes into smaller, more manageable components. Oracle Database offers the widest choice of partitioning methods available, including interval, reference, list, and range. Additionally, it provides composite partitions of two methods, such as order date (range) and region (list) or region (list) and customer type (list). As an option to Oracle Database, Oracle Partitioning is also the foundation of Oracle's Information Lifecycle Management strategy, which aligns the business value of information to cost-effective storage tiers for large data warehousing and transaction processing applications.

  - **OLAP** - The OLAP Option to Oracle Database is a full featured online analytical processing server embedded within the Oracle Enterprise Edition Database. It runs within the kernel of the database, which by default allows it to benefit from standard Oracle Database features such as scalability, reliability, security, backup and recovery, and manageability.
  - **Spatial and Graph** - The Oracle Spatial and Graph option to Oracle Database includes full 3-D and Web Services support to manage all geospatial data including vector and raster data, topology, and network models. It's designed to meet the needs of advanced geographic information system (GIS) applications such as land management, utilities, and defense/homeland security. Oracle's open, native spatial format eliminates the cost of separate, proprietary systems, and is supported by all leading GIS vendors.

- **Oracle Database Gateways** - Oracle Database Gateways address the needs of disparate data access. In a heterogeneously distributed environment, Gateways make it possible to integrate with any number of non-Oracle systems from an Oracle application. They enable integration with data stores such as DB2, SQL Server and Excel, transaction managers like CICS and message queuing systems like WebSphere MQ.
- **Oracle Direct Connector for Hadoop Distributed File System** - Oracle Direct Connector for Hadoop Distributed File System is a high speed connector for accessing data on HDFS directly from an Oracle Database using SQL. Access to the data on HDFS is optimized for fast data movement and parallelized with automatic load balancing. Data on HDFS can be in delimited files or in Oracle data pump files created by Oracle Loader for Hadoop.
- **Oracle Endeca Information Discovery** - Oracle Endeca Information Discovery is an enterprise data discovery platform for rapid, intuitive exploration and analysis of information from any combination of structured and unstructured sources. It enables organizations to extend their existing business analytics investments to unstructured data - such as social media, websites, content systems, files, email, database text and big data. Oracle Endeca Information Discovery provides visibility into any source and supports analysis alongside traditional business intelligence source systems
  - **Endeca Server** - The core search-analytical database organizes complex and varied data from disparate source systems into a faceted data model that is extremely flexible and reduces the need for up-front data modeling.
  - **Content Management System Connectors** - Allows for integration of content stored in various enterprise content management systems such as Documentum, FileNet, Interwoven, Lotus Notes/Domino, Microsoft SharePoint, and OpenText LiveLink.
  - **Text Enrichment (Sentiment Analysis)** - Includes text analysis capabilities for extraction of people, places, organizations, quotes, and aggregate sentiment.
- **Oracle Enterprise Data Quality (OEDQ)** - Oracle Enterprise Data Quality provides organizations with an integrated suite of data quality tools that provide an end-to-end solution to measure, improve, and manage the quality of data from any domain, including customer and product data. Oracle Enterprise Data Quality also combines powerful data profiling, cleansing, matching, and monitoring capabilities while offering unparalleled ease of use.
  - **Oracle Enterprise Data Quality Profile & Audit** - Oracle Enterprise Data Quality Profile and Audit provides a basis for understanding data quality issues and a foundation for building data quality rules for defect remediation and prevention. It provides the ability to understand your data, highlighting

key areas of data discrepancy; to analyze the business impact of these problems and learn from historical analysis; and to define business rules directly from the data.

- **Oracle Enterprise Data Quality Parsing & Standardization** - Oracle Enterprise Data Quality Parsing and Standardization provides a rich palette of functions to transform and standardize data using easily managed reference data and simple graphical configuration. In addition to functions for basic numeric, string, and date fields, functions for contextual data such as names, addresses, and phone numbers are provided.
- **Oracle Enterprise Data Quality Match & Merge** - Oracle Enterprise Data Quality Match and Merge provides powerful matching capabilities that allow you to identify matching records and optionally link or merge matched records based on survivorship rules.

  Matching is a key component of many data quality projects and can be used to support different activities such as de-duplication, consolidation, customer data integration (CDI), and master data management,
- **Oracle Enterprise Data Quality - Product** - In the world of data quality, product data provides some specific challenges. To address these challenges

  To address these challenges Oracle Enterprise Data Quality Product Data Parsing and Standardization uses semantic recognition to quickly recognize the product category and apply the correct rules based on context.

  Product data also presents specific challenges for matching and merging product records. Oracle Enterprise Data Quality Product Data Parsing and Standardization is typically used to create a standardized product record, while Oracle Enterprise Data Quality Product Data Match and Merge is able to identify exact, similar, and related records and optionally merge them based on defined survivorship rules.

- **Oracle Enterprise Repository** - Oracle Enterprise Repository serves as the core element to the Oracle SOA Governance solution. Oracle Enterprise Repository provides a solid foundation for delivering governance throughout the service-oriented architecture (SOA) lifecycle by acting as the single source of truth for information surrounding SOA assets and their dependencies. Oracle Enterprise Repository provides a common communication channel for the automated exchange of metadata and service information between service consumers, providers, policy decision points, and additional governance tooling. It provides the visibility, feedback, controls, and analytics to keep SOA projects on track to deliver business value. The intense focus on automation helps to overcome barriers to SOA adoption and streamline governance throughout the lifecycle.
- **Oracle Essbase** - Oracle Essbase is the industry-leading multi-dimensional OLAP server, providing a rich environment for effectively developing custom analytic and enterprise performance management applications. By leveraging its self-managed, rapid application development capabilities, business users can quickly model complex business scenarios.
- **Oracle Event Processing (OEP)** - Oracle Event Processing is a complete, open architecture that enables the sourcing, processing, and publishing of complex events. This allows filtering, correlation, and processing of events in real-time so that downstream applications are driven by true, real-time intelligence. Oracle Event Processing provides the ability to join incoming streaming events with persisted data, thereby delivering contextually aware filtering, correlation, aggregation, and pattern matching.

- **Oracle Exadata**
  - **Database Machine** - The Oracle Exadata Database Machine provides extreme performance for both data warehousing and online transaction processing (OLTP) applications, making it an ideal platform for consolidating onto grids or private clouds. It is a complete package of servers, storage, networking, and software that is massively scalable, secure, and redundant.
  - **Intelligent Warehouse Solutions** - These are complete data warehousing solutions tailored to address industry-specific BI requirements. They include Oracle Data Model, Oracle Business Intelligence, Oracle Exadata, Oracle OLAP and Oracle Data Mining. Solutions include:
    * Industry-specific advanced and predictive analytics.
    * Enterprise-wide industry aligned logical data models.
    * Pre-built samples of relevant reports and dashboards.

    Note: Since these solutions cover a large portion of the architecture, the Product Mapping View does not include these solutions as a single product, but rather includes the individual products from which they are comprised.

- **Oracle Exalytics** - The Oracle Exalytics In-Memory Machine is an engineered in-memory analytics machine that delivers no-limit, extreme performance for BI and EPM applications. The hardware is a single server that is optimally configured for in-memory analytics for BI workloads and includes powerful compute capacity, abundant memory, and fast networking options.

  The Oracle Exalytics In-Memory Machine features an optimized Oracle BI Foundation Suite (Oracle BI Foundation) and Oracle TimesTen In-Memory Database for Exalytics. The TimesTen In-Memory Database for Exalytics is an optimized in-memory analytic database, with features exclusively available on the Oracle Exalytics platform.
- **Oracle Golden Gate** - Oracle Golden Gate is a comprehensive software package for enabling the replication of data in heterogeneous data environments. The product set enables highly available solutions, real-time data integration, transactional change data capture, data replication, transformations, and verification between operational and analytical enterprise systems.
- **Oracle Loader for Hadoop (OLH)** - Oracle Loader for Hadoop enables data loading from Hadoop into an Oracle database. OLH sorts, partitions, and converts data into Oracle Database formats in Hadoop, and loads the converted data into the database. By preprocessing the data to be loaded as a Hadoop job on a Hadoop cluster, OLH dramatically reduces the CPU and IO utilization on the database commonly seen when ingesting data from Hadoop. An added benefit of presorting data is faster index creation on the data after is has been loaded into the database.
- **Oracle MDM Hubs (OMDM)**
  - **Oracle Customer Hub** - Oracle Customer Hub is a customer data integration (CDI) solution that enables organizations to centralize information from heterogeneous systems, creating a single view of customer information that can be leveraged across all functional departments and analytical systems.

  - **Oracle Supplier Hub** - Oracle Supplier Hub is the application that unifies and shares critical information about an organization's supply base. It does this by enabling customers to centralize all supplier information from heterogeneous systems and thus creates a single view of supplier information that can be leveraged across all functional departments.
  - **Oracle Product Hub** - Oracle product information management data hub is an enterprise data management solution that enables customers to centralize all product information from heterogeneous systems, creating a single view of product information that can be leveraged across all functional departments.
  - **Oracle Site Hub** - Oracle Site Hub is a location mastering solution that enables organizations to centralize site and location specific information from heterogeneous systems, creating a single view of site information that can be leveraged across all functional departments and analytical systems.
  - **Oracle Higher Education Constituent Hub** - Oracle's Higher Education Constituent Hub (HECH) enables higher education institutions to create a complete, authoritative, single view of their constituents including applicants, students, alumni, faculty, donors, and staff. It then feeds the numerous systems on campus with trusted constituent data.
  - **Oracle Data Relationship Management**
    * **Financial Master Data Management** - Create an enterprise view of financial chart of accounts, cost centers and legal entities with a view to govern on-going financial management and consolidation based on consistent definitions of financial and reporting structures across general ledger systems, financial consolidation, planning and budgeting systems.
    * **Analytical Master Data Management** - Create an enterprise view of analytical dimensions, reporting structures, performance measures and their related attributes and hierarchies using Oracle Hyperion Data Relationship Management's data model-agnostic foundation. Construct departmental perspectives that bear referential integrity and consistency with master data constructs based on validations and business rules that enforce enterprise governance policies. Synchronize master data with downstream systems including business intelligence (BI)/enterprise performance management (EPM) systems, data warehouses and data marts to gain trustworthy insight.

- **Oracle NoSQL Database** - Oracle NoSQL Database is built using Oracle Berkeley DB Java Edition High Availability as the underlying, storage system. It includes ACID transactions, persistence, high availability, high throughput, data distribution, dynamic partitioning, predictable latency, and lights out administration. Oracle NoSQL Database is designed and tested to work with the rest of the Oracle technology stack. Oracle NoSQL Database is installed and configured as part of the Big Data Appliance. There are already proven integration points with Oracle Data Integrator, Oracle Loader for Hadoop and Oracle Complex Event Processing.
- **Oracle R Connector for Hadoop** - enables interactive access to Hadoop data from R.
- **Oracle Real Time Decisions** - combines both rules and predictive analytics to power solutions for real-time enterprise decision management. It enables real-time intelligence to be instilled into any type of business process or customer interaction.

- **Oracle Secure Enterprise Search (OSES)** - Oracle Secure Enterprise Search provides a familiar user interface to internet search users and gives secure access to all of an organization's data sources-Web sites, file servers, content management systems, enterprise resource planning and customer relationship management systems, business intelligence systems, and databases. Oracle Secure Enterprise Search provides better access to enterprise information, while protecting sensitive data from unauthorized users.
- **Oracle Service Bus** - Oracle Service Bus is a proven, lightweight and scalable SOA integration platform that delivers low-cost, standards-based integration for high-volume, mission critical SOA environments. It is designed to connect, mediate, and manage interactions between heterogeneous services, legacy applications, packaged applications and multiple enterprise service bus (ESB) instances across an enterprise-wide service network. Oracle Service Bus provides built-in management and monitoring capabilities and supports out-of-the-box integration with SOA Governance products.
- **Oracle Service Registry** - Oracle Service Registry provides a 'DNS'-like reference for SOA. A fully compliant UDDI v3 registry, Oracle Service Registry provides a standards-based interface for SOA runtime infrastructure to dynamically discover and bind to deployed service end points. Oracle Service Registry bridges the gap between the design time and runtime environments through automated synchronization with Oracle Enterprise Repository and Oracle SOA Suite.
- **Oracle SQL Developer Data Modeler** - SQL Developer Data Modeler is a data modeling and design tool, proving a full spectrum of data and database modeling tools and utilities, including modeling for Entity Relationship Diagrams (ERD), Relational (database design), Data Type and Multi-dimensional modeling, with forward and reverse engineering and DDL code generation. The Data Modeler imports from, and exports to, a variety of sources and targets, provides a variety of formatting options and validates the models through a predefined set of design rules.
- **Oracle WebCenter Content** - Oracle WebCenter Content provides organizations with a unified repository to house unstructured content, and deliver it to business users in the proper format, and within context of familiar applications to fit the way they work.

7.2 Product Mapping

The following sections illustrate the mapping of Oracle products in relation to the scenarios documented in the logical view chapter.

Some products have multiple "significant" capabilities and therefore may appear in more than one scenario. Some product capabilities may not be shown due to space considerations.

The components and interactions are logical representations, not actual product architecture diagrams. Please consult Oracle product documentation for a complete list of features, architecture documentation, product usage, and integration capabilities.

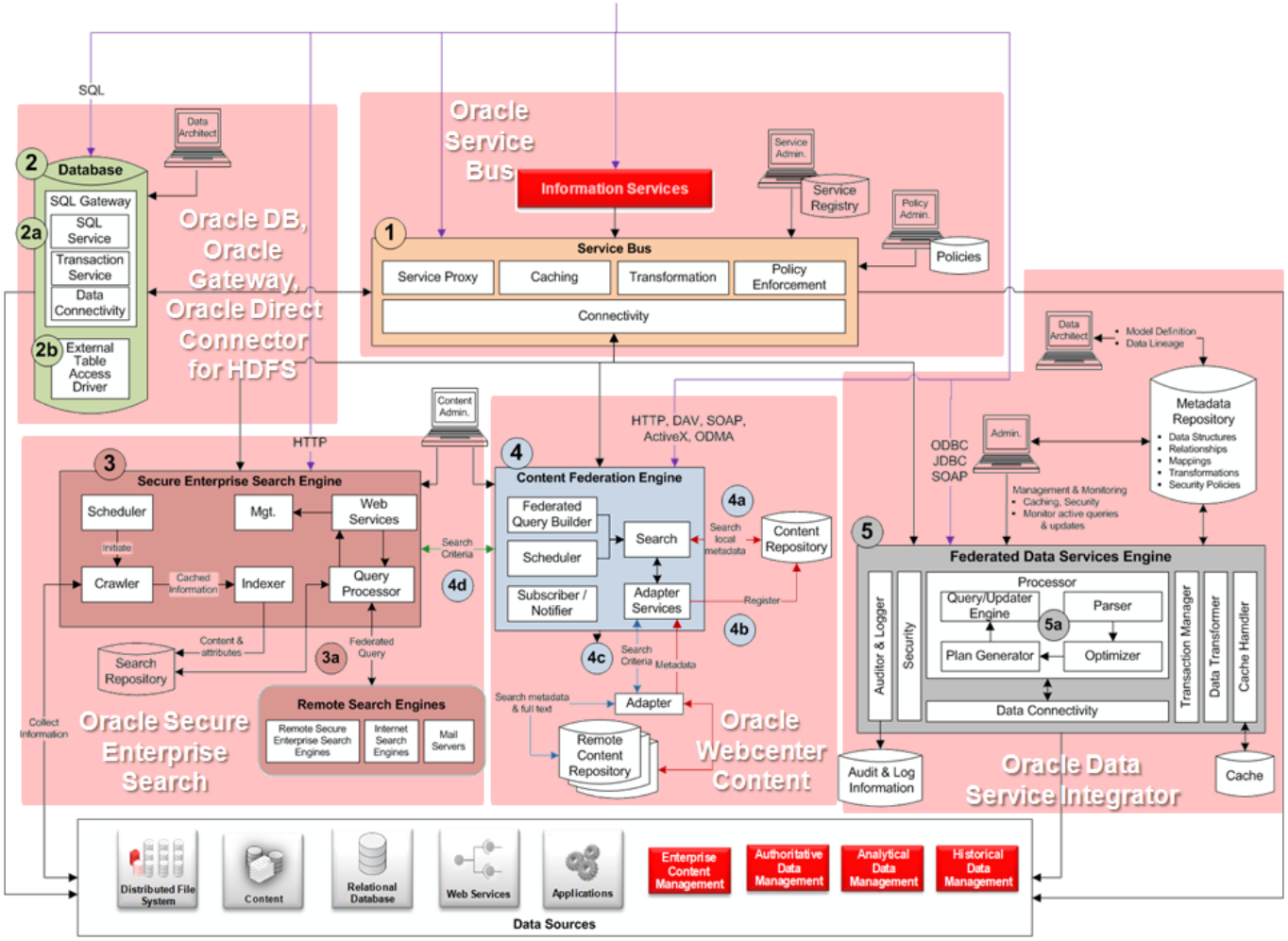
Security and management products are also not shown. These mappings can be found in the *ORA Security* and *ORA Management and Monitoring* documents. Consuming tools and applications are not shown. For some example product mappings in this area refer to the *ORA Business Analytics Infrastructure* document.

7.2.1 Information Consumption

7.2.1.1 Virtualization

Figure 7–1 depicts Oracle products as they relate to components of the virtualization logical view highlighted in the previous chapter.

Figure 7–1 Virtualization - Product Mapping



Oracle has a comprehensive range of products that can be utilized as data sources such as relational databases, in-memory databases, NoSQL databases, (distributed) file systems, repositories, and packaged/bespoke horizontal/vertical applications.

Oracle has a number of engines that support the virtualization architecture.

Oracle Database Gateway (OGW) utilizes an Oracle database to integrate with any number of non-Oracle systems by providing a gateway to other databases and providing a virtual view. It handles the SQL request and performs the necessary semantic translation before accessing the underlying data store(s). Data appears to be integrated in a single virtual database while actually remaining in its current distributed locations. This approach is useful when organizations want to perform ad-hoc queries or updates on infrequently accessed data that is more appropriately located elsewhere.

Oracle Direct Connector (ODC) for HDFS is a high speed connector for accessing data on HDFS directly from an Oracle Database. ODC for HDFS gives users the flexibility of accessing and importing data from HDFS at any time, as needed by their application. It allows the creation of an external table in an Oracle Database, enabling direct SQL access on data stored in HDFS. The data stored in HDFS can then be queried via SQL, joined with data stored in Oracle Database, or loaded into the Oracle Database. Access to the data on HDFS is optimized for fast data movement and parallelized, with automatic load balancing. Data on HDFS can be in delimited files or in Oracle data pump files created by Oracle Loader for Hadoop.

Oracle Data Services Integrator (ODSI) provides a federated bidirectional (read and write) Data Services approach which enables access of disparate data stores via a services interface. The ODSI engine resolves any semantic conflicts by supporting a model driven approach to mapping and transformation. This approach is appropriate when there are few joins and small result-sets. In addition to supporting a services interface, the ODSI engine also supports ODBC/JDBC, allowing reporting and querying tools to access the engine natively without any knowledge of the underlying data stores.

Oracle Secure Enterprise Search (OSES) provides a virtual and secure access to pre-crawled content in local and remote enterprise search engines. OSES provides a familiar http user interface to internet search users and gives secure access to all of an organization's data sources-Web sites, file servers, content management systems, enterprise resource planning and customer relationship management systems, business analysis systems, and databases.

Oracle Webcenter Content (OWC) utilizes federated queries of local and remote content repositories simultaneously utilizing standardized APIs from a single point of access. The results of the federated queries are aggregated and returned according to the specified criteria, e.g. results are ordered by date creation. These queries encompass both textual and metadata searches. It is also possible for OWC to re-direct the request to Oracle's Secure Enterprise Search engine which has already performed a scheduled crawl of content. This allows for content to be included in the search that the local and remote content repositories are unaware of.

In addition to the engine highlighted above, Oracle also offers another engine called Oracle BI Server. Oracle BI server is a highly scalable, highly efficient query reporting and analysis server that provides services to support virtualization. This is covered in more detailed in ORA Business Analytics Infrastructure document.

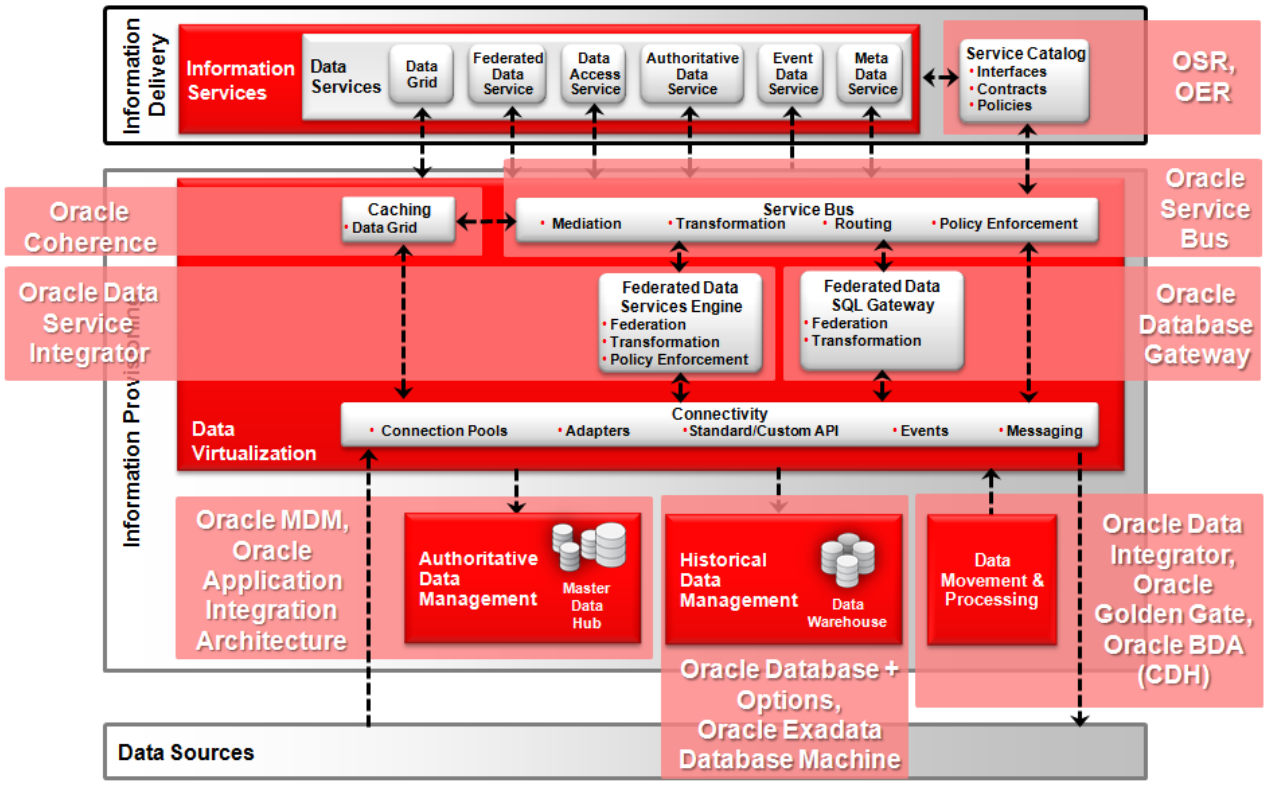
While each engine supports direct access, it is advantageous to access the engine via Oracle's Service Bus (OSB). It provides the service endpoint and mediation capabilities. OSB is used to decouple service consumers from service providers via a service proxy. Service providers can change over time transparently to consumers. The service bus can also handle message transformation, connectivity, and security.

There are several areas where Oracle Engineered Systems can be utilized for virtualization. Oracle Exadata and Oracle Database appliance align well as data stores and SQL gateways. The remaining products in this scenario can be hosted by both Oracle Exalogic and SPARC Super Clusters.

7.2.1.2 Information as a Service

Figure 7–2 depicts Oracle products as they relate to components of the Data Services scenario highlighted in the logical view chapter.

Figure 7–2 Data Services - Product Mapping



Data Services provide a single point of access for aggregated, real-time, operational and historical data. Data Services operate on, and expose, a common business information model. This is a logical semantic data model that is backed by one or more physical data sources. The purpose of this model is to provide consumers with a clear semantic view of data irrespective of physical data model characteristics.

Potential service consumers can browse or search the repository for services that match their needs and download these artifacts to begin the service engineering process. See *ORA SOA Infrastructure* for more details regarding Oracle Service Registry (OSR) and Oracle Enterprise Repository (OER).

Data Services are aware of constructs such as relations and objects. This allows Data Services to be as simple as:

- Fetching data from a database
- Event driven data state notification
- Fetching data from a data grid such as Oracle Coherence (OC)
- Performing federated data source queries (via Oracle Database Gateway or Oracle Data Services Integrator) in real time with aggregated data result sets.

In addition, Data Services can enable transactional create, update or delete operations to an underlying data store while maintaining business and referential integrity rules.

These Data Services leverage the virtualization capabilities of Oracle Service Bus (OSB). OSB provides the service endpoint and mediation capabilities. The service bus is used to decouple service consumers from service providers via a service proxy. Service providers can change over time transparently to consumers. The service bus can also handle message transformation, connectivity, and security. The service bus then routes the request to the appropriate engine, e.g. Oracle MDM Hub/Oracle AIA,

Oracle Database, and Oracle Data Integrator. In addition, Data Events are also supported. Whether they originate from Oracle's Database or from within Oracle Data Integrator, these events can be captured and routed to the appropriate subscribing Data Event Service.

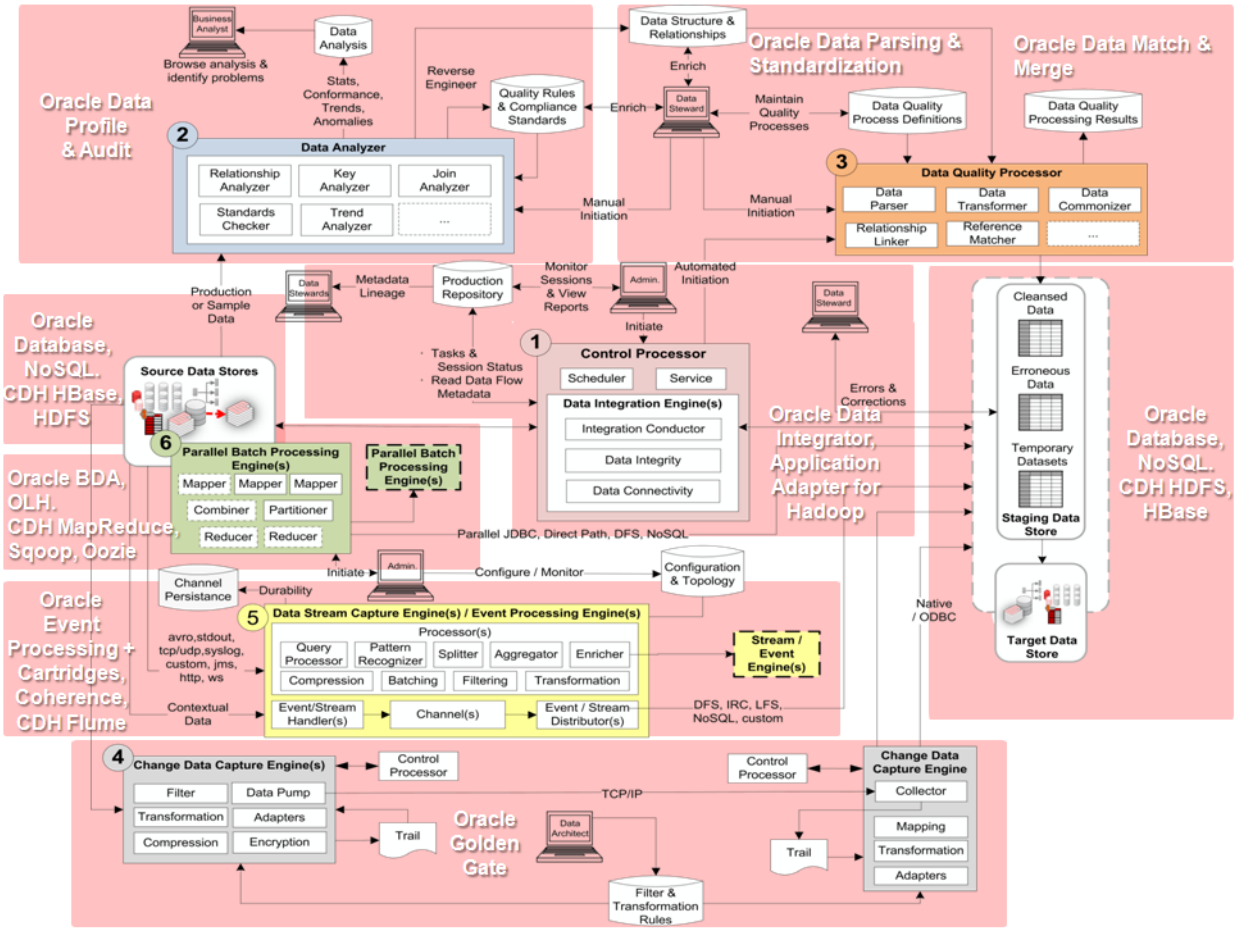
Oracle has a comprehensive range of products that can be utilized to develop, test and deploy Data Services (See *ORA Engineering* document). Oracle products mentioned in this chapter, such as Oracle MDM Hub and Oracle ODI, have out-of-the-box Data Services functionality.

7.2.2 Information Acquisition

7.2.2.1 Ingestion

Figure 7–3 depicts Oracle & Cloudera products as they relate to components of the Ingestion logical view highlighted in the previous chapter.

Figure 7–3 Ingestion - Product Mapping



Oracle has a number of engines that supports overall ingestion architecture.

Oracle Event Processing (OEP) provides an open architecture for sourcing, processing, and publishing complex events throughout the enterprise. OEP can handle a high throughput of events with micro-second latency while at the same time evaluating

patterns in event data. It utilizes Oracle Coherence (OC) for integrated caching and data grid technology to scale to enterprise-class event processing use cases.

Events can be captured by the event producer no matter which format or protocol is used. While OEP supports standards such as JMS and HTTP, it also supplies an adapter layer that can mediate differences in format and protocol. OEP provides a rich, declarative environment based on Oracle Continuous Query Language (Oracle CQL), a query language based on SQL with added constructs that support streaming data, to improve the efficiency and effectiveness of managing business operations. Once processed events can then be distributed via several standards based means. Again an adapter layer is supplied that can mediate the difference in format and protocol for the target of the event(s).

Oracle Event Processing can be extended through the use of data cartridges. Data cartridges enable the ability to integrate domain data types and functions with the Oracle CQL language, allowing the usage of these extensions within Oracle CQL queries in the same way you use Oracle CQL native types and built-in functions. Below are some of the available data cartridges.

- **Hadoop/NoSQL Cartridge** - Provides access to Hadoop and NoSQL via CQL queries. This enables the enrichment of events with data from NoSQL and Hadoop.
- **Spatial Cartridge** - Provides the ability to perform the most important geographic domain operations such as storing spatial data, performing proximity and overlap comparisons on spatial data, and integrating spatial data with the Oracle Event processing server by providing the ability to index on spatial data
- **JDBC Cartridge** - Provides the ability to execute a SQL query against a database and use its returned results in a CQL query.
- **Java Cartridge** - Provides the ability to write Oracle CQL queries and views that seamlessly interact with the Java classes in your Oracle event application.

In effect, OEP is a complete solution for building applications to filter, correlate and process events in real-time so that downstream data processing/analytics, applications, service oriented architectures and event-driven architectures are driven by true, real-time intelligence.

Above HDFS sits a framework called Apache MapReduce for parallel processing of large data sets across a large number of nodes. Computational processing can occur on data stored either in a file system (unstructured) or in a database (structured). MapReduce can take advantage of locality of data, processing data on or near the storage assets to decrease transmission of data.

Apache Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts log and event data by utilizing streaming data flows. Even though multiple targets are supported the most common destination for the data is HDFS.

Apache Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases. Sqoop can be utilized to import data from external structured datastores into Hadoop Distributed File System or related systems like Hive and HBase. In addition, Sqoop can be used to extract data from Hadoop and export it to external structured datastores such as relational databases and enterprise Data Warehouses. If the data target is an Oracle database then it is preferential to utilize Oracle's Loader for Hadoop dramatically as it reduces the CPU and IO utilization on the database commonly seen when ingesting data from Hadoop. An added benefit is faster index creation on the data after is has been loaded into the database.

Oracle Data Integrator (ODI) delivers next-generation declarative design batch and real-time integration by utilizing Extract Load and Transform (E-LT) technology that improves performance, reduces data integration costs, even across heterogeneous systems. It achieves this by leveraging disparate relational database management systems engines to process and transform the data. This approach optimizes performance and scalability and lowers overall solution costs. In effect ODI generates native code for disparate RDBMS engines (SQL, bulk loader scripts, for example). E-LT architecture extracts data from sources, loads it into a target, and transforms it using the database power. By leveraging the database, Oracle Data Integrator reduces network traffic by transforming data in the database containing the target table. Therefore the E-LT architecture delivers the highest possible performance.

As well as being a data integration engine, ODI can utilize and orchestrate other engines by generating native code for engines (e.g. MapReduce) and executing system utilities remotely.

Oracle Loader for Hadoop (OLH) enables data loading from Hadoop into an Oracle database. OLH sorts, partitions, and converts data into Oracle Database formats in Hadoop and loads the converted data into the database. By preprocessing the data to be loaded as a Hadoop job on a Hadoop cluster, Oracle Loader for Hadoop dramatically reduces the CPU and IO utilization on the database commonly seen when ingesting data from Hadoop. An added benefit of presorting data is faster index creation on the data after is has been loaded into the database.

ODI Application Adaptor for Hadoop provides native Hadoop integration within Oracle Data Integrator. Based on Hive, it enhances productivity and provides a simple user interface for creating programs to load data to and from files or relational data stores. Hadoop implementations require complex Java MapReduce code to be written and executed on the Hadoop cluster. Using ODI and the ODI Application Adapter for Hadoop, developers use a graphical user interface to create these programs. Utilizing the ODI Application Adapter for Hadoop, ODI generates optimized HiveQL which in turn generates native MapReduce programs that are executed on the Hadoop cluster. For example, ODI with this adaptor can be used to build Hadoop metadata within ODI, load data into Hadoop, transform data within Hadoop, and load data easily and directly into Oracle Database utilizing Oracle Loader for Hadoop.

Oracle Golden Gate (OGG) is a comprehensive software package for enabling the replication of data in heterogeneous data environments. OGG can address incremental data ingestion requirements through transactional change data capture (CDC). The changed records are captured from the source data stores via a CDC engine and then routed, transformed, and delivered to the target CDC engine, which in turn populates the target data store.

To support data quality requirements, the products under the banner Oracle Enterprise Data Quality can be utilized. Oracle Data Quality Profile & Auditing (ODP) profiles the source data and determines the appropriate rules that are required to bring the data into compliance. These rules are utilized by Oracle Data Quality Parsing and Standardization (ODPS) to enable the transformation and standardization of data. Oracle Enterprise Data Quality Match and Merge (ODMM) provides powerful matching capabilities that allow the identification of matching records and optionally link or merge matched records based on survivorship rules. This is required when activities such as de-duplication and consolidation are required as part of a data quality initiative.

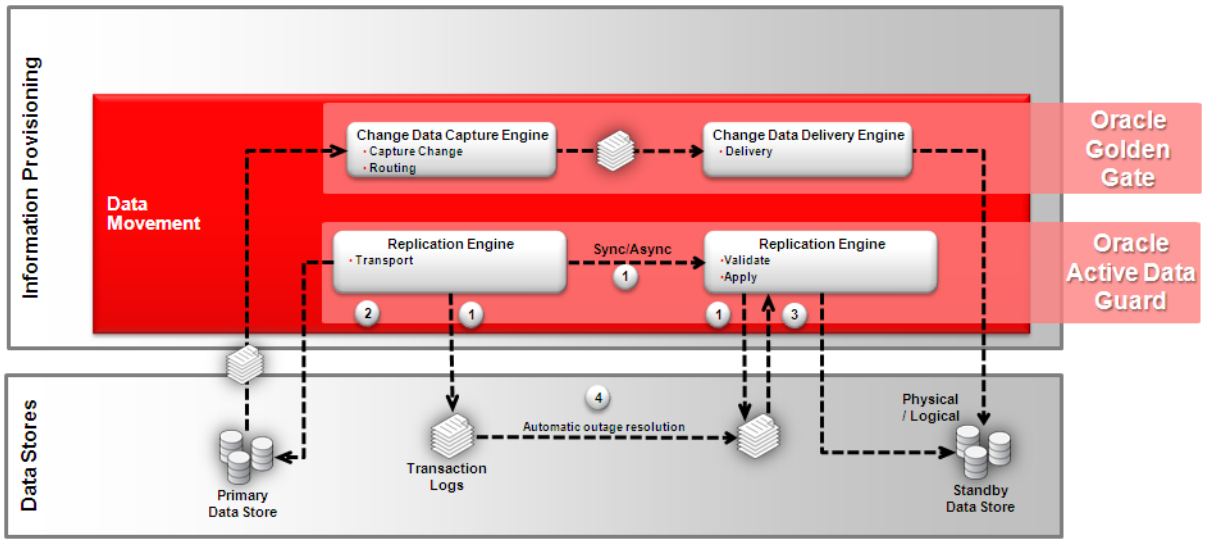
There are a number of areas where Oracle Engineered Systems can be utilized for ingestion. For example, Oracle Big Data Appliance including Cloudera Distribution of Hadoop, Exadata Oracle Exadata, and Oracle Database Appliance align well as data

stores and transformation engines. Oracle Exalogic can be used to host various ingestion engines such as the event processing engine.

7.2.2.2 Replication

Figure 7–4 depicts Oracle products as they relate to components of the replication logical view highlighted in the previous chapter. Replication enables the creation and maintenance of one or more synchronized standby data stores that protect data from failures, disasters, errors, and corruptions.

Figure 7–4 Replication - Product Mapping



Oracle Golden Gate (OGG) is a comprehensive software package for enabling the replication of data in heterogeneous data environments. OGG enables high availability and disaster recovery solutions, through data replication and verification utilizing a transactional change data capture (CDC). The changed records are captured from the source data stores via a CDC engine and then routed and delivered to the target CDC engine which in turn populates the target data store. While Oracle's Active Data Guard (OADG) is utilized as a replication engine for homogeneous source and target databases.

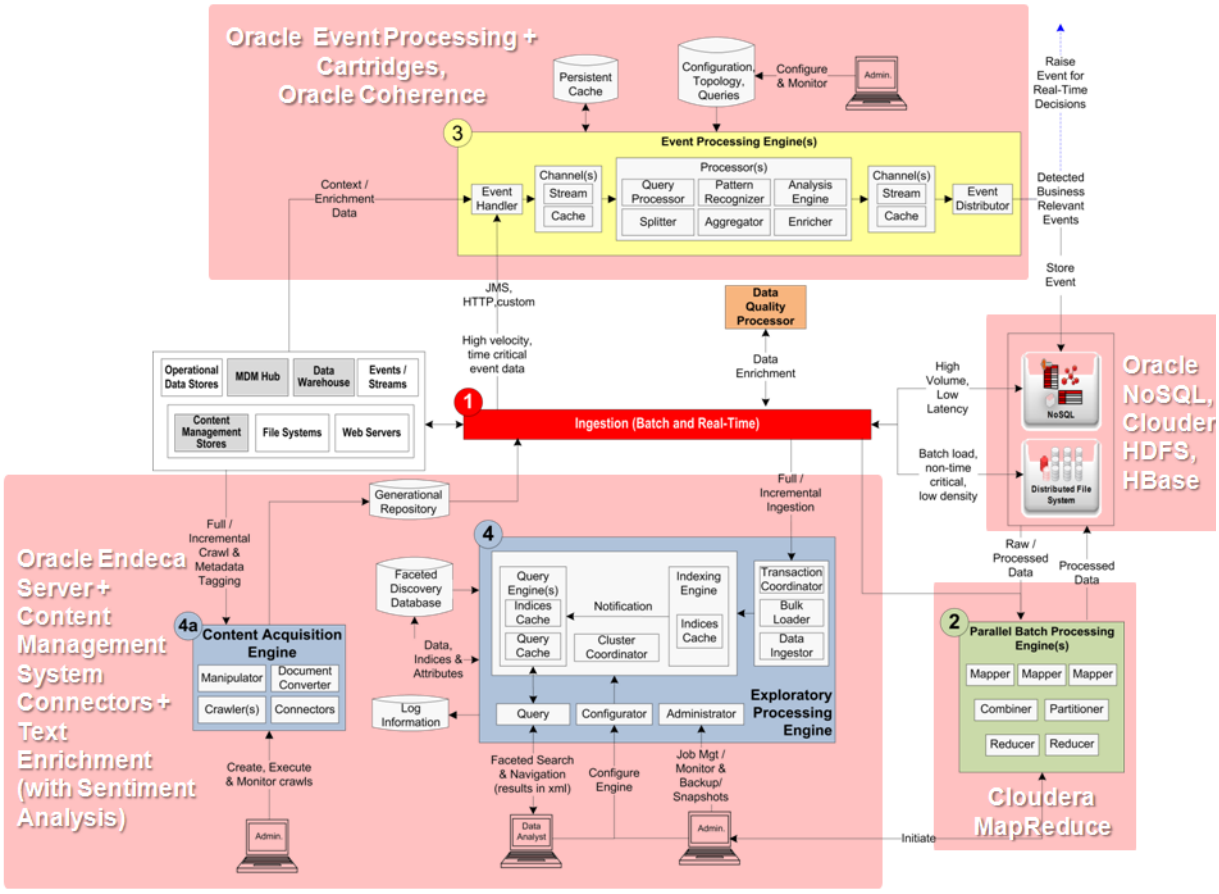
Refer to Oracle's Maximum Availability Architecture (MAA) for more details.

7.2.3 Information Organization

7.2.3.1 Big Data Processing

Figure 7–5 depicts Oracle products as they relate to components of the big data processing scenario highlighted in the logical view chapter.

Figure 7–5 Big Data Processing - Product Mapping



Oracle offers a comprehensive range of products that can be utilized as data stores such relational databases, in-memory databases, and file systems. But for Big Data Processing, the most prevalent tend to be NoSQL databases, and distributed file systems.

Refer to Section 7.2.2.1, "Ingestion" for product mapping details with regards to ingestion.

Oracle Endeca Server is a hybrid, search/analytical database that provides unprecedented flexibility in combining diverse and changing data as well as extreme performance in analyzing that data. Oracle Endeca Server has the performance characteristics of in-memory architecture however is not memory bound. Oracle Endeca Server's data model is based on the concept of "records". Roughly analogous to Data Warehouse facts, each record consists of an arbitrary collection of attributes made up of key/value pairs. These attributes can have a single value, multiple values, or a hierarchy of values based on an external taxonomy definition. Values may be text of any length, numeric, date/time, geospatial, or Boolean.

Oracle Endeca Server does not employ an overarching schema for data. Instead, every record has its own schema. This enables the rapid integration of any type of data, whether structured or unstructured, from inside or beyond the confines of the enterprise Data Warehouse, without the efforts associated with traditional relational data modeling. Oracle Endeca offers both an exploratory processing engine as well as a content acquisition engine as part of its integration suite. This suite includes a set of

tools used for loading, and optionally enriching, diverse information including structured, unstructured, and semi-structured content into Oracle Endeca Server

Oracle Endeca offers a number of add-on modules:

- **Content Management System Connectors** - These connectors allows for the integration of content stored in various enterprise content management system such as Documentum, FileNet, Interwoven, Lotus Notes/Domino, Microsoft SharePoint, and OpenText LiveLink.
- **Text Enrichment** - This add-on module includes text analysis capabilities for extraction of people, places, organizations, quotes, and themes as well as core summarization capabilities for automatic creation of abstracts or topical summaries.
- **Text Enrichment with Sentiment Analysis** - This add-on module includes all of the Text Enrichment capabilities as well as advanced text analysis for extraction of aggregate sentiment related to each extracted concept. Sentiment is extracted with a score indicating the positive and negative nature of a document or an entity in the document. These scores can be used within the faceted search experience by varying ranges of positivity and negativity.

While not shown in the diagram, Oracle's Endeca product includes a studio application. This is a discovery application composition environment for the Oracle Endeca Server. Studio provides drag and drop authoring to create highly interactive, visually-rich, enterprise-class information discovery applications. Refer to *ORA Business Analytics Infrastructure* document for more details on Endeca studio.

Oracle Event Processing and its various cartridges enable the processing and manipulation of event data through the application of business rules to identify opportunities, threats, and anomalies that need to be acted upon. Capabilities such as pattern recognition, splitting, filtering, correlation, aggregation, and transformation are utilized to eventually generating the outbound stream of events.

While Oracle Event Processing product can perform a level of big data processing, (e.g. early alerting), it is common for additional and more comprehensive processing to also take place on a less time-sensitive basis utilizing Parallel Batch Processing Engine such as Cloudera MapReduce. The use of Oracle's Event Processing Engine as part of Big Data Processing is to immediately filter out irrelevancies and identify patterns of interest that can be used to an organization's advantage or to defuse an immediate business threat.

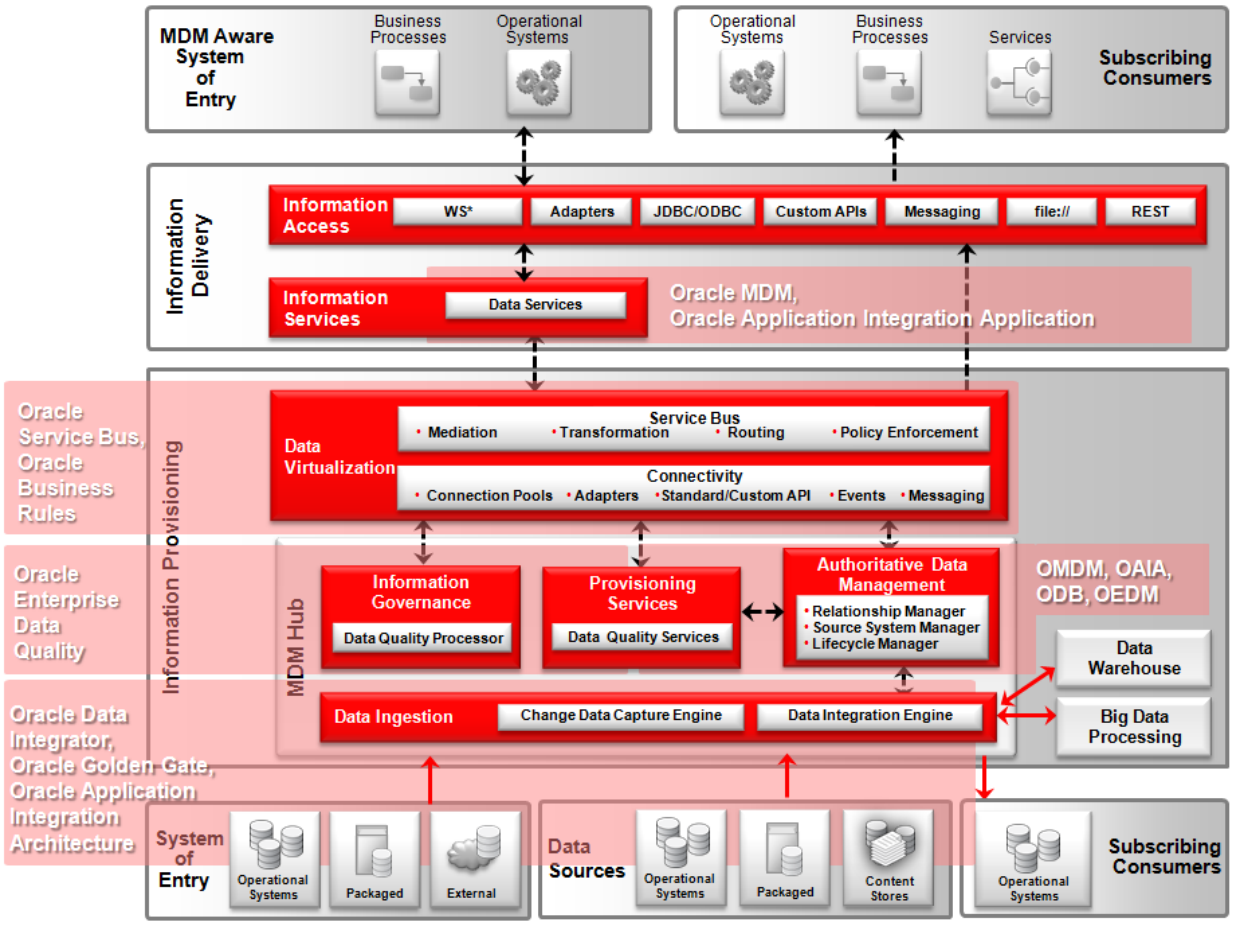
The current popular choice for parallel batch data processing is a programming model called MapReduce, which addresses the challenges arising from the scale of the data (e.g. volume, velocity), the varied formats (unstructured and semi-structured), and the complex techniques needed to process it. Oracle offers Cloudera's Distribution of Hadoop which offers MapReduce and HDFS as well as many other open source projects.

There are several areas where Oracle Engineered Systems can be utilized for big data processing. The majority of the focus is on Oracle Big Data Appliance. Oracle Big Data Appliance brings Big Data solutions to mainstream enterprises. Built using industry-standard hardware from Sun and Cloudera's Distribution including Apache Hadoop, Big Data Appliance is designed and optimized for Big Data workloads. By integrating the key components of a Big Data platform into a single product, Oracle Big Data Appliance delivers an affordable, scalable and fully supported Big Data infrastructure without the risks of a custom built solution. Oracle Big Data Appliance integrates tightly with Oracle Exadata and Oracle Database using Oracle Big Data Connectors, and seamlessly enables analysis of all data in the enterprise, both structured and unstructured.

7.2.3.2 Enterprise Master Data Management

Figure 7–6 depicts Oracle products as they relate to components of the master data management scenario highlighted in the logical view chapter.

Figure 7–6 Master Data Management - Product Mapping



While it is possible to build your MDM hub from scratch using Oracle products, Oracle offers a number of pre-built MDM Hub solutions. Oracle's MDM Hub solutions (OMDM) include the applications that consolidate, clean, govern, and share the master operational data.

Oracle Application Integration Application (OAIA) delivers a composite application framework utilizing Foundation Packs and Process Integration Packs (PIPs) to support real time synchronous and asynchronous events that are leveraged to maintain quality master data across the enterprise.

Oracle Golden Gate (OGG) and Oracle Data Integrator (ODI) offer a comprehensive and high performance data synchronization infrastructure that understands MDM dimensions, hierarchies and cross-reference files. This is essential for MDM in that it loads, updates and helps automate the creation of MDM data hubs. MDM also utilizes data integration for integration quality processes and metadata management processes to help with data lineage and relationship management.

This makes ODI the perfect choice for 1) loading MDM Data Hubs, 2) populating Data Warehouses with MDM generated dimensions, cross-reference tables, and hierarchy information, and 3) moving key analytical results from the Data Warehouse to the MDM Data Hubs to 'operationalize' the information.

Oracle MDM applications seamlessly work with Oracle's Service Bus (OSB) and are familiar with the data structures and access methods of Oracle's MDM Hubs. OSB provides the service endpoint and mediation capabilities and enables the decoupling of service consumers from service providers via a service proxy. Oracle MDM maintains an integration repository to facilitate Web Service discovery, utilization, and management. The service bus can handle message transformation, connectivity, security, and routing. Its routing algorithms have access to Oracle's Business Rules (OBR) Engine and can coordinate message traffic with data quality rules established by corporate governance teams in conjunction with the MDM Hubs.

While the MDM tables can exist on a standalone Oracle Database (ODB), there are other options and advantages. The OLTP MDM tables can coexist with the Data Warehouse on Oracle's Exadata Database machine (OEDM) moving data around a disk array, creating summarized tables, and utilizing materialized views, without offloading terabytes of data through costly networks to multiple hardware platforms. Availability and scalability are maximized even as the total cost of ownership is reduced..

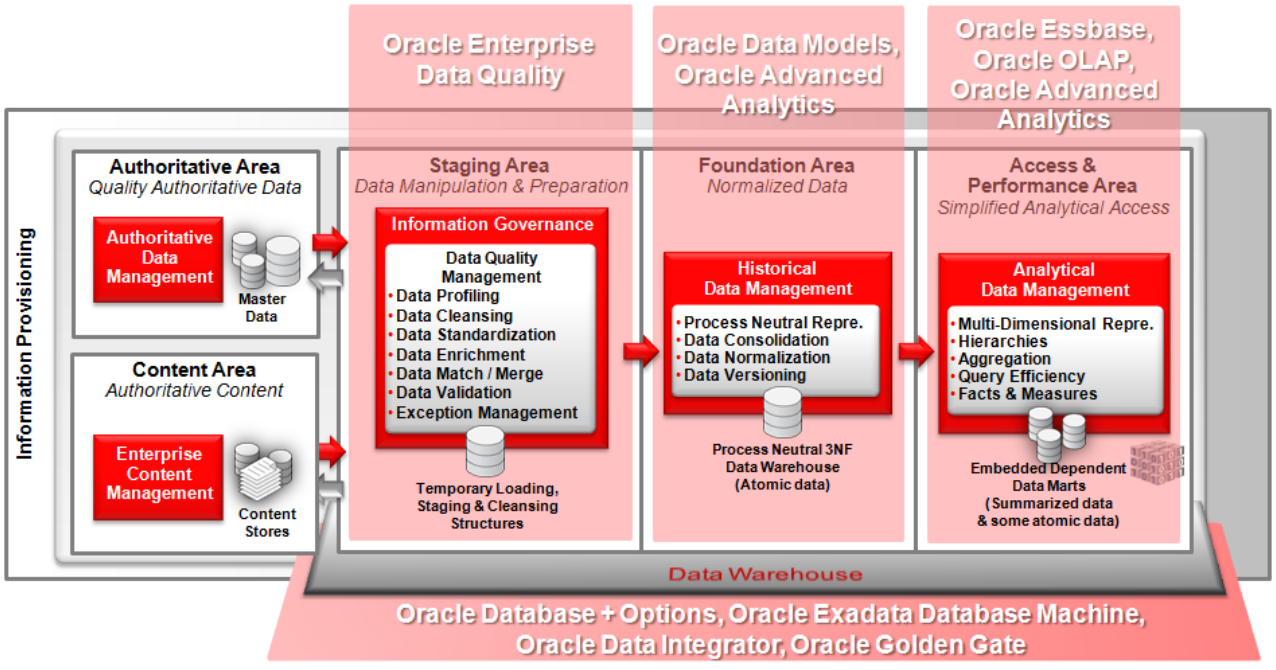
While not shown in the above figure, an Event Driven Architecture for real-time complex event processes can be essential for triggering appropriate MDM related business processes when key data items are changed.

There are several areas where Oracle Engineered Systems can be utilized for virtualization. Oracle Exadata and Oracle Database Appliance align well as data stores and SQL gateways. The remaining products in this scenario can be hosted by both Oracle Exalogic and SPARC Super Clusters.

7.2.3.3 Data Warehouse

Figure 7–7 depicts Oracle products as they relate to components of the hybrid hub and spoke data warehouse highlighted in the logical view chapter.

Figure 7–7 Data Warehouse - Product Mapping



The entire Data Warehouse can be implemented using Oracle Database (ODB) and various options. Oracle Exadata Database Machine (OEDM) can also be utilized as it provides extreme performance for both data warehousing and online transaction processing (OLTP) applications, making it an ideal platform for consolidating onto grids or private clouds. It is a complete package of servers, storage, networking, and software that is massively scalable, secure, and redundant. For more details consult the *ORA Engineered Systems* document.

Data are acquired from various data sources and are assumed to be un-cleansed, unfiltered, and potentially incomplete. Data can originate from both internal and external sources. Oracle's Data Integrator (ODI) and Oracle Golden Gate (OGG) are key components in data ingestion from various data sources into the Staging Area. Depending on the deployment model, ODI, OGG, and/or ODB can be utilized to move data between the areas.

Once the data has been cleansed with Oracle's Enterprise Data Quality (OEDQ) product, it can be moved into the Foundation Area. The Foundation Area represents the heart of the Data Warehouse and is the area responsible for managing the data over the long term including versioning and history. It is advantageous to leverage an enterprise information model sourced from one of the industry bodies or an enterprise data warehousing model from database vendors such as Oracle or a specialist model vendor. Oracle offers a number of data models which can be used, but some adaptation of the logical model to meet local needs is typically required.

Data mining capabilities within Oracle Advanced Analytics (OAA) allow large quantities of data to be mined without the need to transfer that data from the database to another server for processing. OAA is highlighted in both the Foundation Area and the Access & Performance Area.

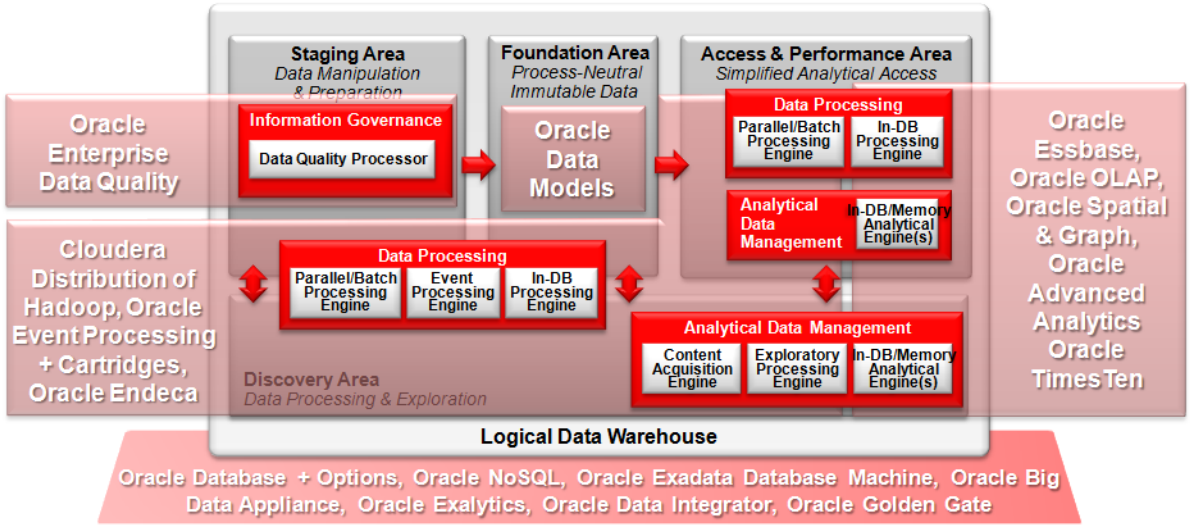
The Access and Performance Area adds the access components to the architecture and enables responsive queries via several mechanisms such as indices, materialized views, columnar data storage, or in-memory data management. In addition, either Oracle OLAP or Oracle Essbase can be utilized in delivering OLAP functionality.

For more details regarding the Access and Performance area consult the *ORA Business Analytics Infrastructure* document.

7.2.3.4 Logical Data Warehouse

Figure 7–8 depicts Oracle products as they relate to components of the Logical Data Warehouse (LDW), highlighted in . Section 6.3.4.4, "Logical Data Warehouse"

Figure 7–8 Logical Data Warehouse - Product Mapping



The Logical Data Warehouse brings together two architectures (big data processing, Data Warehouse). For more product mapping details refer to the product mapping section for these architectures.

To manage structured data within the LDW, Oracle Database (ODB) can be implemented with various options. Options such as OLAP and Spatial & Graph can be used to manage all aspects of the LDW that pertain to structured data. The OLAP option is used to provide a dimensional/cube view, while the Spatial & Graph option adds geospatial and network data models.

For unstructured data, Oracle NoSQL database, Oracle Endeca, and Cloudera's Distribution of Hadoop (CDH) are generally utilized. Oracle NoSQL database can be used as a distributed, high performance key/value store. It is most useful when records need to be stored as one opaque value that can be access via an assigned key. It supports real-time create, read, update, and delete operations. CDH, namely the Hadoop Distributed File System (HDFS), is used as a distributed data store for Hadoop applications. Oracle Endeca is used to associate and correlate all types of data - both structured and unstructured. It will often be used in the Discover Area as a means to model collections of data in different ways and to perform root cause analysis

Not all data in an LDW will be required to be cleansed. But if it is, then can be cleansed with Oracle's Enterprise Data Quality (OEDQ) product where the cleansed data can then be moved into the Foundation Area.

The Foundation Area is responsible for managing the data over the long term including versioning and history. For structured data, it is advantageous to leverage an enterprise information model sourced from one of the industry bodies or an enterprise data warehousing model from database vendors such as Oracle or a specialist model vendor.

Three engineered systems have been identified as components of the LDW architecture: Oracle Exadata, Oracle Exalytics, and Oracle Big Data Appliance.

Oracle Exadata is specifically designed to run the Oracle Database (including options such as OLAP, Spatial & Graph, and Advanced Analytics. It can house multiple Oracle Databases in a single rack-mount system. Oracle Exadata offers tremendous performance benefits over standalone deployments, based on its high performance hardware, network, and high capacity memory and disk.

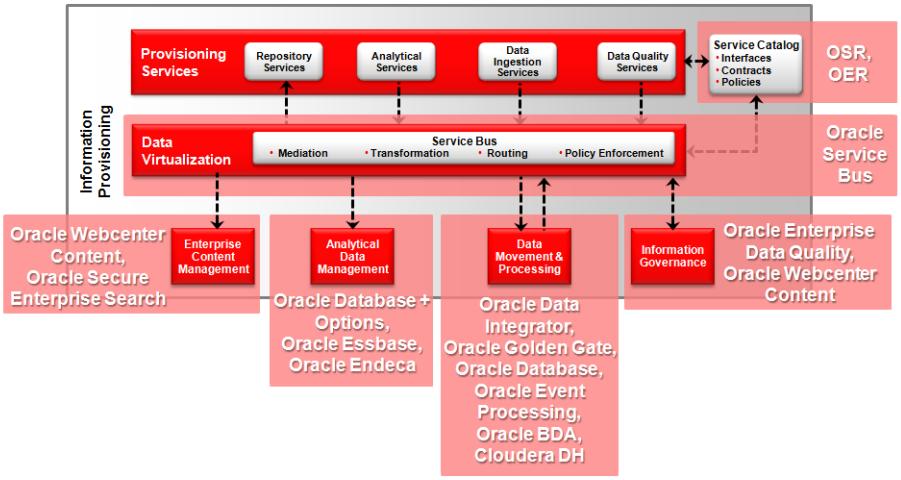
Oracle Exalytics is designed for high performance analytics. It combines Oracle hardware with an optimized Oracle BI Foundation Suite and Oracle TimesTen In-Memory Database for Exalytics. OBIFS consists of several products including Oracle Essbase and the Oracle BI Server. Therefore, Oracle Exalytics is a deployment option that provides both Essbase OLAP capabilities as well as information virtualization.

Oracle Big Data Appliance is Oracle's integrated suite of hardware and software designed for Big Data management and processing. It includes Cloudera's Distribution of Hadoop (CDH).

7.2.3.5 Information Provisioning as a Service

Section 7–9, "Provisioning Services - Product Mapping" depicts Oracle products as they relate to components of Provisioning Services, as highlighted in Section 6.3.5, "Information Provisioning as a Service".

Figure 7–9 Provisioning Services - Product Mapping



The implementation of Provisioning Services, enables the logical abstraction of underlying architectural concerns, and presents the provisioning of both operational and analytical information via a clearly defined, pre-integrated, and simplified interface.

Potential consumers of provisioning services can browse or search Oracle's design-time enterprise repository (OER) for details regarding Provisioning Services that match their needs and download these artifacts. To dynamically discover and bind to deployed Provisioning Service end points, Oracle Service Registry (OSR) can be utilized. See *ORA SOA Infrastructure* document for more details regarding OSR and OER.

Provisioning Services are services that expose highly optimized engines for the provisioning and distribution of data. Therefore, the underlying architectural concerns of the Information Provisioning Layer have been abstracted and presented in a clearly defined interface.

While Provisioning Services can be directly accessed, the logical view utilizes Oracle's Service Bus (OSB) to leverage its virtualization capabilities, with the service request being routed to the appropriate information provisioning engine.

8

Deployment View

This section provides an example of how the information management products, mapped in the previous chapter, might be deployed.

A number of factors influence the way products are deployed in an enterprise (e.g. load and high availability requirements, network traffic, hardware resources). These factors will influence decisions about the number of physical machines to use for each product. Federation and disaster recovery concerns will influence the number of deployments and failover strategy to use.

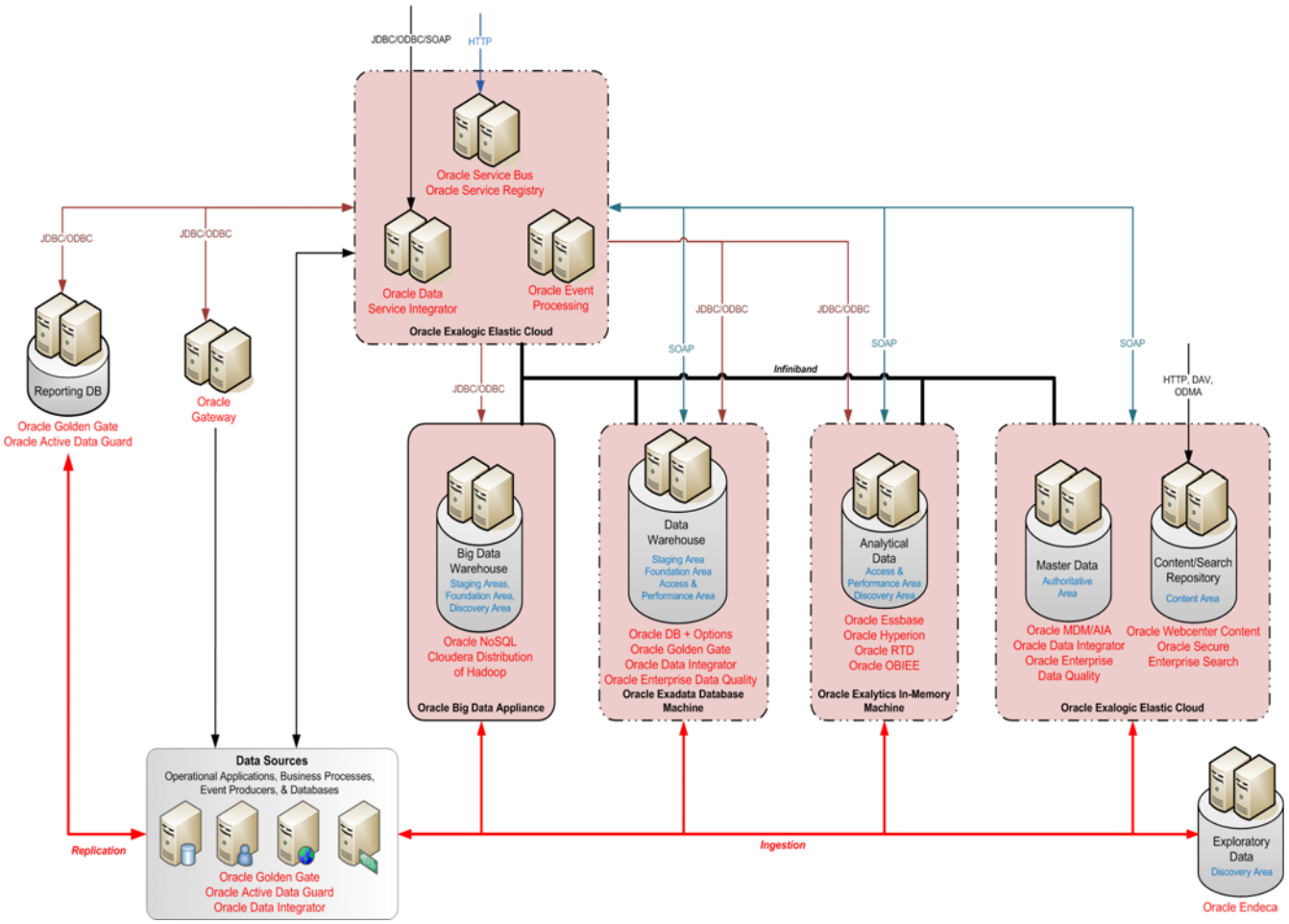
Oracle offers several engineered systems. The engineered systems represent an integrated and optimized hardware and software solution. The solutions, often incorporating multiple software products, represent a deployment option that differs from a standalone, distributed deployment strategy.

In addition, deployment configurations and options may vary depending on product versions. Given these and other variables, it is not feasible to provide a single, definitive deployment for the products, or to guarantee the deployment scenarios presented in this chapter will work for a particular product version.

8.1 Sample Information Management Deployment

Figure 8–1 presented here is used as an example.Please consult product documentation and Oracle's Maximum Availability Architecture (MAA) for further deployment information.

Figure 8–1 Example Deployment



Central to the deployment is the use of Oracle's Service Bus (OSB) which supports multiple protocols for both incoming and outgoing requests. The OSB provides several capabilities including routing, transformation, and protocol mediation. The service bus connects to various information management infrastructure components. Even though not shown in the diagram, information consumers can bypass the service bus and go directly to each infrastructure component, but these information consumers will lose the flexibility that the service bus offers.

With the service bus being so important it should be designed to be highly available, scalable, and secure. Clustering allows OSB to run on a group of servers that can be managed as a single unit. In a clustered environment, multiple machines share the processing load. OSB provides load balancing so that resource requests are distributed proportionately across all machines. In addition, clustering provides high availability. OSB supports failover for clustered objects so that it is able to recover from server failures.

Oracle Data Service Integrator and Oracle Event Processing are shown deployed on clustered Oracle Weblogic Servers and therefore inherit the performance, scalability, fault tolerant aspects of an industry leading JEE container.

Oracle Service Bus, Oracle Data Service Integrator, and Oracle Event Processing could be optionally deployed together on Oracle Exalogic Elastic Cloud. Exalogic is a highly scalable, open-ended deployment platform that is designed for applications based on Oracle Fusion Middleware (FMW). It comes with the Oracle VM, Oracle Enterprise

Manager, and Oracle Virtual Assembly Builder. Exalogic is an option for virtually any FMW-based application.

Oracle Gateway is show on its own server independent of any Oracle or non-Oracle database. Depending on network considerations, the Gateway has the option to be on the same machine as an Oracle database or on the same machine as the 3rd party Server database.

As detailed earlier in the document, it is common for organizations to replicate data from operational systems so that their reporting needs do not affect the performance of the operational systems. The above diagram highlights data from several types of data stores are replicated to a set of reporting databases using a variety of products, e.g. Oracle Golden Gate, Oracle Active Data Guard.

The Oracle NoSQL Database is shown, along with Cloudera's Distribution of Hadoop (CDH), deployed on Oracle Big Data Appliance. Currently, Oracle only offers CDH as a feature of Oracle Big Data Appliance.

While it is possible to deploy the layers of a Data Warehouse over several servers and databases, there are obvious advantages to having all of the layers centrally deployed. The speed of data ingestion between the layers will be an essential element in providing a more dynamic Data Warehouse. There are many advantages to utilizing engineered systems to address the needs of Data Warehousing. Separate physical machines have traditionally been used in order to avoid a single point of failure. However, modern hardware options, such as Oracle Exadata, can be used to accomplish high availability in a single physical machine. This is supported using redundant processors, power supplies, network cards, fans, etc. that can be hot swapped without taking the equipment out of service. Oracle Exadata also enhances scalability and performance via high-speed processors, multiple cores, large cache and memory sizes, and a high performance internal network. See the *ORA Engineered Systems* document for more details.

Oracle Essbase and a number of additional Oracle analytical applications are shown deployed on Oracle Exalytics In-memory Machine. Refer to the *ORA Business Analytics Infrastructure* document for deployment details.

Oracle WebCenter Content and Oracle MDM products are also shown optionally deployed on Oracle's Exalogic Elastic Cloud.

Another area of consideration is the possibility of dividing up network traffic for specific purposes. For instance, a client access network provides connectivity to the Information Services and Data Virtualization capabilities. While a management network is utilized to administer various infrastructure components and addresses any denial of service concerns. Another option is to have a Data ingestion network (red arrows) where the ingestion and/or replication of data can be separated from the client access so that quality of service can be maintained. This data ingestion network could utilize communications between Oracle's engineered systems via a very high speed network called InfiniBand. Each InfiniBand link provides 40 Gigabits of bandwidth - many times higher than traditional storage or server networks.

9

Summary

To be competitive, today's businesses are required to make informed operational and analytical decisions in a timely manner. Information is key to enabling this competitive advantage. However, many businesses are struggling to achieve pervasive access to information on which to base critical business decisions.

Standard reporting no longer addresses the needs of the business as the volume, velocity, variety, and variability of information increases. Organizations need to extend their existing business analytics investments to unstructured sources - including social media, websites, content systems, files, and email, therefore providing unprecedented visibility into the business, saving time and cost, and leading to better business decisions.

To achieve the benefits that Big Data Processing and Business Analytics has to offer, organizations will have to reassess their Information Governance programs, information acquisition capabilities, as well as their overall enterprise information management strategy. Characteristics such as appropriate data quality, ingesting data from disparate sources, and normalized master data play a pivotal part in alleviating many issues that organizations will encounter when executing an overall information management strategy.

If businesses are to truly deliver information pervasively throughout a business and beyond, then an information management platform is required which must be capable of loading and querying data in near real-time with the same level of accuracy and high availability as is expected from every other operational system.

With ever growing data volumes and deeper analysis requirements, it also follows that an information management platform must be able to scale out to meet future demands and manage the information lifecycle to reduce costs in a realistic and secure platform.

This document has detailed aspects of an information management reference architecture, which in combination with other initiatives such as SOA, and Business Analytics can form the foundation of an information management framework. This reference architecture must be customized and extended for your environment to take into account your business goals, business conditions and existing IT environment.

Deploying a comprehensive information platform requires a strategic approach that cannot be achieved by a single project. Therefore, it is imperative that an information management platform must be seen as a longer term vision that can be achieved over a series of coordinated pragmatic projects.

At the same time, it must be understood that the information management platform is seen as a means to an end. Delivering timely, secure, and accurate information to the appropriate information consumers so that they can make informed operational and analytical decisions should be the goal.

A

Further Reading and References

The following references provide more information on information management and related IT architecture topics.

A.1 ITSO Related Documents

This document is part of a series of documentation and supporting material called IT Strategies from Oracle (ITSO), designed to enable organizations to develop an architecture-centric approach to enterprise-class IT initiatives. The following ITSO documents have been referenced throughout this document.

- *ORA Business Analytics Infrastructure* plays a key role in a successful enterprise Business Analytics environment. This document describes the role of infrastructure and the capabilities it provides for Business analytics. It offers an array of views to define infrastructure for Business Analytics, including logical and physical views, as well as technology and product mapping
- *ORA SOA Infrastructure* plays a key role in a successful enterprise SOA environment. This document describes the role of infrastructure and the capabilities it provides for SOA. It offers an array of views to define infrastructure for SOA, including logical and physical views, as well as technology and product mapping.
- *ORA EDA Infrastructure* plays a key role in a successful enterprise EDA environment. This document describes the role of infrastructure and the capabilities it provides for EDA. It offers an array of views to define infrastructure for EDA, including logical and physical views, as well as technology and product mapping.
- *ORA Integration* - Many forms of integration exist today and play a key role in enterprise computing. The ORA Integration document examines the most popular and widely used forms of integration, putting them into perspective with current trends made possible by SOA standards and technologies. It offers guidance on how to integrate systems in the Oracle Fusion environment, bringing together modern techniques and legacy assets.
- *ORA Security* - The ORA Security document describes important aspects of security including identify, role, and entitlement management, authentication, authorization, and auditing (AAA), and transport, message, and data security.
- *ORA Management and Monitoring* - This document provides a reference architecture for designing a management and monitoring framework to address the needs for the modern IT environment.

- *ORA Engineered Systems* - This document delves into the reasons the industry is moving towards engineered systems, how they impact a customer's architecture, and how they impact both enterprise and cloud architectures.

The above documents are available on Oracle's Technology Network.

A.2 Other Resources and References

There are a large number of additional resources that have either been referenced or serve as a basis for further reading.

- Enterprise Data Services in SOA using ODI (Oracle, February 2009)
- Oracle Endeca Information Discovery - A Technical Overview (Oracle, 2012)
- Data Warehouse Architectures (Hugh J. Watson, Thilini Ariyachandra, 2005)
- The Data Management Body of Knowledge (DAMA-DMBOK, 2009)
- Building the Data Warehouse (W.H. Inmon, Wiley)
- The Data Warehouse Lifecycle Toolkit (Ralph Kimball et al, Wiley)
- Various W.H. Inmon and Ralph Kimball Articles
- Hadoop: The Definitive Guide (Tom White, 2011)
- Enterprise Integration Patterns (Gregor Hohpe, Bobby Wolf, Addison-Wesley, 2003)
- Oracle Information Integration, Migration, and Consolidation (Jason Williamson, Tom Laszewski, Prakash Nauduri, Packt Publihsing, 2011)
- Enabling Pervasive BI through a Practical Data Warehouse Reference Architecture. (Doug Cackett, Andrew Bond, Kevin Lancaster, and Keith Laker. Oracle White Paper. February 2010.)
- Reference Architecture: Information Management & Big Data. (Doug Cackett, Andrew Bond, John Gouk - Oracle 2013)

Lastly refer to Oracle product documentation which contains a wealth of information on this topic.