

The Origins of Business Intelligence

The origins of business intelligence may be traced back to the first data processing applications, which were simple applications such as accounts payable and receivable. These applications ran on sequential technology, such as magnetic and paper tapes. Using sequential media for storage meant the entire file had to be accessed, even if only a fraction of the file was needed. Oxide often stripped off of magnetic tapes, and entire files were lost. These issues led to the need for a new way to analyze information.

In this chapter we discuss how enterprises tackled the challenges of extracting data from online transaction processing systems, dealing with poor data quality, and structuring the data. We describe what a data warehouse is from its data model, table structure, granularity, and support of historical data, and how it fits into a broader intellectual concept called the corporate information factory, discussed later in the chapter.

Early Data Processing Applications

With sequential storage, data was organized onto what was called a master file, which held central information that was useful to many applications. Punch cards, magnetic tapes, and reports were generated from the applications, but were soon replaced by disk storage. With disk storage, data could be accessed directly and efficiently. Processors grew more powerful and versatile, and the speed and costs of processing dropped dramatically. As data was stored onto disk, master files mutated into databases. A

database is a centralized collection of data that resides on disk storage and is available for processing by any application that needs it.

With disk storage, transactions could be processed directly and online against the central databases that were built. This meant that the transaction could execute in a short amount of time—from 2 to 3 seconds. Soon, online transaction processing resulted in online applications. Online applications were tied together by the central online databases that they ran against.

Online applications focused on high availability and good response time. The online transaction application soon became central to running the business, as the online transaction became an essential part of direct interaction with the customer. However, the custom development of online transaction applications led to several challenges:

- Aging of applications as soon as they were put online
- Sheer number of applications
- No documentation
- New requirements
- Inability to change the system once developed
- Fragility of the system
- Sensitivity to response time

An additional problem with many online applications was lack of integration among them. Each online application was developed according to its own specification, and a different set of requirements shaped each one. There was no common understanding of what was:

- A customer
- A product
- A transaction
- A vendor
- A shipment

Because each application was so difficult to change, no significant reinterpretation could be accomplished. The result was that even the largest and most sophisticated corporation did not know who its customers were. A corporation would spend a huge amount of money each month on technology without knowing the most basic information, such as who are the best customers, what products are selling, and how much revenue was made last quarter.

Enter Extract Files

The first reaction to the challenge of not having corporate data was to create what was called an *extract file*. An extract file would be created by a database from one application and shipped to another application, so it seemed that data could be shared and corporate data could be created. Extracts became very popular, and soon there were a

lot of them. Every new extraction exacerbated the problems of the spiderweb. Adding extractions made matters worse, not better. The problems of the spiderweb included:

Data integrity. The same element of data appeared in many places. In one place the element had a value of 25. In another place the element had a value of 67. In still another place the element had a value of 135. No one really knew what the right value was.

Data redundancy. The sheer redundancy of data was enormous. The same data was being shuffled from one place to the next, and in doing so, the burden of massive amounts of data being repeated over and over began to add up to significant amounts of storage and processing power.

Timeliness of data. While being shuffled around the system, data was aging. In one day, the value of a unit of data may change five or six times. The extract processing simply was not capable of keeping up with the speed with which data changed.

Multiple silo of data were created. Each silo was its own operating domain with no coordination or integration with outside silos. The organization found itself making decisions that were contrary to the interest of other parts of the organization.

The extract processing froze an already moribund system. Online transaction applications were difficult to change in any case. But wrapping lines of extraction around the online applications glued those applications into a permanent position.

Data became much more inaccessible. The extract processing placed coordination requirements on the environment, which ensured that accurate data was impossible to obtain, and so forth.

Of particular interest is the lack of historical data. Online applications value current data. How much is a bank account balance right now? Where is a shipment right now? What is the status of an insurance claim right now? Online applications optimize the “right now” aspect of information processing. As soon as data became dated, it was discarded. Lots of historical data clogged the arteries of efficient online processing. Therefore, online data and processing required that older data be jettisoned as soon as possible.

But there is real value in historical data. With historical data, organizations can start to see the forest *and* the trees. With historical data, organizations can start to understand their customer base, because customers are creatures of habit.

Because there was no corporate integrated data or historical data, data was difficult to access. Even if accessed, data was not trustworthy, so it is no wonder organizations began to grow frustrated with their ability to find and process information. Department after department would say, “I know the data is somewhere in my corporation; if I could only get at it.”

The frustrations of the end user with data locked in the spiderweb environment resulted in the realization that there were different kinds of data. There was an essential difference between *operational data* and *informational data*. Table 1.1 outlines those differences.

Table 1.1 Characteristics of Operational versus Informational Systems

OPERATIONAL	INFORMATIONAL/DSS
Detailed	Summarized
Can be updated	Snapshot records; no updates allowed
Accurate up to the second	Timestamp on each record
Used for clerical purposes	Used by management
Built based on requirements	Built without knowing requirements
Supports small uniform transactions	Supports mixed workload
Yields 2- to 3-second response time	Yields 30- to 24-hour response time
Data designed for optimal storage	Data designed for optimal access
Very current data	Mainly historical data
Data is application oriented	Data is integrated
Data designed around functional usage	Data designed around subject areas
Referential integrity is useful	Referential integrity is not useful
High availability is normal	High availability is nice to have

A fundamental split exists between *operational information* and *informational information*. Operational information is used to support the daily operations of a business. Informational information is commonly called *decision support systems* (DSS) information. The foundation for DSS processing became an architectural structure known as the data warehouse. A *data warehouse* is a physically distinct place from the online operational application. In the following sections we describe the data warehouse and how it enables informational information.

What Is a Data Warehouse?

Since the beginning of movement toward data warehousing, data warehouses have been defined as being:

Subject-oriented. Data is organized around a major object or process of an organization. Classic examples include subject area databases for customer, material, vendor, and transaction.

Integrated. The data from various subject areas should be rationalized with one another.

Nonvolatile. Data in a data warehouse is not updated. Once a record is properly placed in the warehouse, it is not subject to change. This contrasts with a record of data in an online environment, which is indeed very much subject to change.

Time-variant. A record is accurate only as of some moment in time. In some cases the moment in time is a single moment. In other cases it is a span of time. But in any case, the values of data found in a data warehouse are accurate and relevant only to some moment in time.

Created for the purpose of management decisions.

The preceding definition has remained unchanged since the inception of the data warehouse. In addition, the data warehouse provides:

- Detailed or granular data
- Integrated data
- Historical data
- Easy-access data

The data warehouse is at the center of the business intelligence environment. The data warehouse represents the single version of truth for the corporation and holds data at a granular level. In addition, the data warehouse contains a robust amount of historical data. The need for a data warehouse is as true within the confines of SAP as it is outside of SAP. And the elements of a data warehouse are as valid for SAP as for the non-SAP environment.

The data warehouse evolves from these requirements and supports the process of moving data from source systems, transforming, and cleansing the data so that it may be stored in an integrated data model at an atomic level of granularity. There are many factors that influence the design of a data warehouse and the structure that data records are stored. We discuss some of these factors in the next sections.

The Data Model

The design of the data warehouse begins with a data model. At the highest level, the data model is known as an *entity relationship diagram* (ERD). The ERD represents the abstraction of the granular data found in the data warehouse. Note that for the purposes of data warehouse design the ERD represents only granular data, not derived data. This distinction is important because it greatly limits the size and complexity of the data model. There are, of course, other data models outside of the data warehouse environment that do attempt to take into account derived data and atomic data.

The ERD consists of entities and relationships. Each entity represents a major subject area of the corporation. Typical subject areas are customer, product, transaction, and vendor. Each entity is further defined at a lower level of data modeling called the *data item set* (DIS). The DIS specifies a lower level of detail than the entity does, encompassing such things as keys and attributes, as well as the structure of those things. The DIS is further broken down into a low level of design called the *physical design*. At the physical level of design the physical characteristics of the data are created.

NOTE For a more detailed description of the methodology required to build a data warehouse, visit the Web site www.billinmon.com.

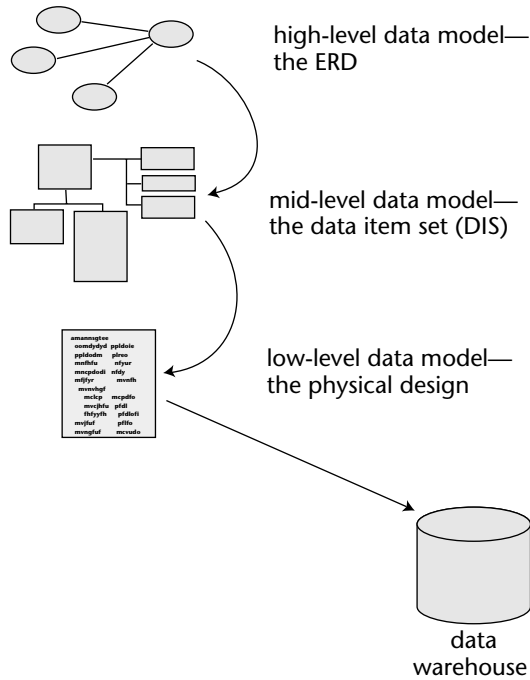


Figure 1.1 The data warehouse is designed from the data model.

The data warehouse is now specified and defined to the database management system that will house it. Other physical aspects of database design such as partitioning, loading, indexing, storage media, and timestamping are determined here as well. Figure 1.1 shows the design of the data warehouse from the different components of the data model.

Different Physical Tables

The data warehouse is made up of interrelated tables or physical databases. Within the data warehouse, there are different physical tables that represent different subject areas or even subsets of subject areas. One table relates to another by means of a shared key or foreign key relationship. The data warehouse has five subject areas:

- Customer
- Product
- Shipment
- Vendor
- Order

Each subject area resides on a separate physical table or database. Collectively the different tables along with their relationships form a data warehouse.

Integration and Transformation Processing

One of the most important and most difficult aspects of data warehouse development and population is that the movement and conversion of data from the operational/legacy source environment. It is estimated that for the first iteration of development, at least 75 percent of the resources required for development will be expended here. During extraction, data is pulled from the legacy environment and moved into the data warehouse environment. This data is pulled from a variety of sources, such as mainframe order entry systems, proprietary shop flow control systems, and custom-built payroll systems.

But data is not merely moved from the legacy environment to the data warehouse environment. Instead, data undergoes a thorough transformation as it is moved, including:

- Converting data into a common format
- Reformatting data
- Realigning encoded values
- Restructuring data
- Assigning default values
- Summarizing
- Resequencing
- Converting keys
- Converting from one database management system (DBMS) to another
- Converting from one operating system to another
- Converting from one hardware architecture to another
- Merging different record types
- Creating meta data that describes the activities of conversion
- Editing data
- Adding a timestamp

In the early days of data warehousing there was no way to create the interface programs between the legacy environment and the data warehouse other than to write them by hand. But with recent technology, extract/transfer/load (ETL) software automatically creates the interfaces needed to bring the data into the world of data warehouse. Figure 1.2 shows the place of ETL between the source systems and the data warehouse.

One of the real benefits of ETL processing is that data enters the ETL process in an application mode and exits in an integrated corporate mode. The processing that occurs inside ETL software allows the data to be integrated.

SAP BW addresses the ETL process by including a so-called Staging Engine to perform simple transformations and conversions. In Chapter 6 we detail the function delivered within the SAP BW related to ETL, as well as discuss the usage of specialized ETL tools that are certified with SAP BW.

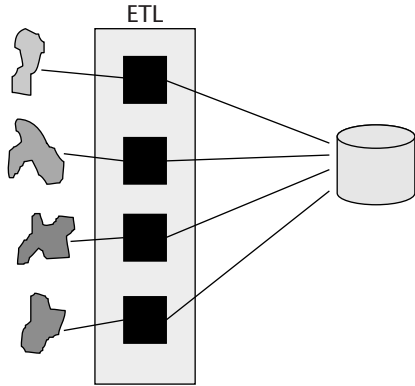


Figure 1.2 Integration and conversion is performed by ETL processing.

Granular Data

The data found in a data warehouse is very granular. This means that the data is placed in the data warehouse at a very low level of detail. The data may then be reshaped by an application so that it can be viewed in a distinct manner.

Sometimes called the atomic data of the corporation, granular data makes up the “single version of truth” that is at the basis of reconciliation for informational processing. Having the granular data at the core of the data warehouse provides many benefits. A primary advantage is the same data can be viewed in different ways. Figure 1.3 shows that marketing looks at data one way, sales looks at it another way, and finance yet another way. But all three departments have a single source of reconcilability.

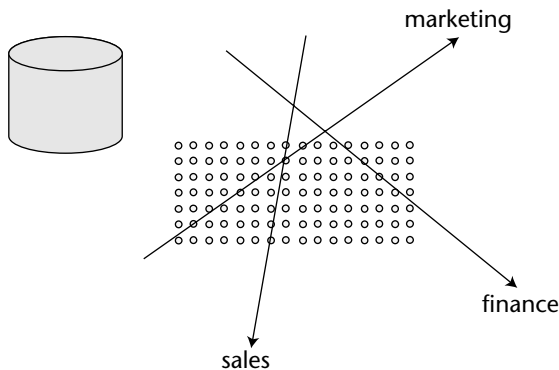


Figure 1.3 Granular data allows the same data to be examined in different ways.

Usually each grain of information in the data warehouse represents some finite unit of measure or business activity for the corporation. For instance, a grain of information might represent details of the following:

A sale. The amount, the date, the item sold, the location of the sale, or the customer

An order. The date of the order, the product ordered, or the amount of the order

A telephone call. The time of the call, the length of the call, the calling party, or the person called

A delivery of a product. The date of the delivery, the location of the delivery, or the person making the delivery

Each grain of information can be combined with other grains to provide a different perspective of data.

In addition to allowing data to be viewed differently by different parties, another benefit is that the granular data may lie in wait in the data warehouse for unknown and future requirements. Then when a requirement becomes known, the granular data can be shaped immediately to suit the new requirements. There is no need to go to the operational/legacy environment and pull data out. This means that the data warehouse puts the corporation in a proactive rather than reactive position for new needs for information.

Historical Data

One of the most important characteristics of a data warehouse is that it contains a robust amount of historical data. Figure 1.4 shows a data warehouse that contains 5 years of history. Such an amount of history is typical. However, some warehouses may contain even more historical data and other data warehouses may contain less data, depending on the business needs of the corporation.

Although historical data has many applications, perhaps the most potent is the ability to step backward in time and do what-if analysis. Doing so allows you to gain insights that cannot be achieved any other way.

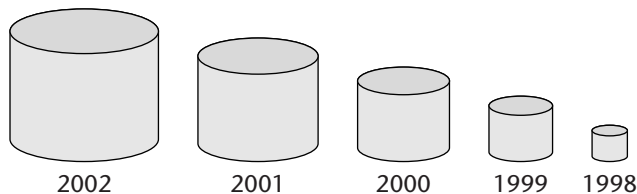


Figure 1.4 The data warehouse contains robust amounts of historical data.

Timestamping

The units of data stored inside the data warehouse are timestamped so that each unit of data in the data warehouse has some element of time associated with the record. The timestamping of data warehouse data signifies that the unit of data is accurate as of the timestamp.

In general, there are two ways that a record is stored in the data warehouse: discretely or continuously (see Figure 1.5). In a discrete record there is one instant in time for which the record is accurate. In a continuous record, there is a span of time for which the record is accurate. These records form a larger definition of information over time.

Usually discrete records are used for a large number of fast-changing variables. Continuous timestamps are used for a small number of variables that change slowly and for which there is value in knowing information over time.

The records in a data warehouse have a distinctive structure, including:

- A timestamp
- A key
- Primary data
- Secondary data

Data Relationships

The different types of data found in the data warehouse relate to each other by means of foreign keys pointing to actual keys. As an example, suppose customer ABC places an order. There would be a customer record for customer ABC, as well as a separate order record for the order. The order record, in its body, would have a foreign key reference to customer ABC.

The data relationships found in the data warehouse are special in that they are delimited by time. When a relationship is indicated in the data warehouse, the relationship is intended to only be valid for the moment in time indicated by the timestamps found on the participating records. This interpretation of a relationship is quite different from that of referential integrity found in the online environment.

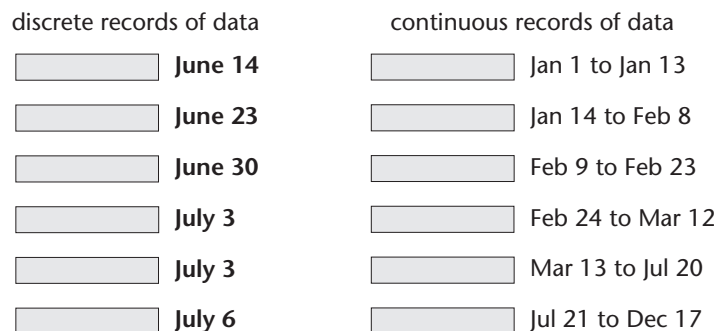


Figure 1.5 Timestamped records are either continuous or discrete.

Generic Data versus Specific Data

One design issue that arises in every data warehouse is how to account for generic data and specific data at the same time. Generic data applies to all instances of a subject area. Specific data applies to only certain occurrences of a subject area.

The generic database stores customer information, along with related tables, including a wholesale customer table, a European customer table, a long-term customer table, and a preferred customer table. Each of the outlying tables contains information specific to only the class of tables that meet the criteria. For example, a preferred wholesale customer would have data in the generic customer table, in the preferred customer table, and in the wholesale customer table.

In such a manner, data of different types can be represented efficiently in a data warehouse.

Data Quality

Data quality is an important issue for the data warehouse environment. As shown in Figure 1.6, there are three places where data quality is addressed:

- At the point of data entry to the legacy/operational environment
- At the point of ETL processing
- Once the data resides inside the data warehouse itself

For instance, raw data entry is addressed inside the legacy/operational environment. The problem is that the budget for doing tasks such as maintenance here has long gone away. In addition, no shop is anxious to go poking around old, fragile, undocumented legacy applications lest something untoward and unexpected happen. Therefore, not much data quality activity occurs in the legacy/operational environment.

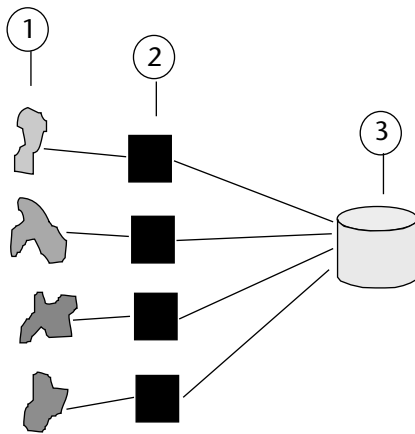


Figure 1.6 Base data for all customers is kept in one table; specific data for different types of customers is kept in separate, unique tables.

Most data quality activity occurs at the moment of ETL. ETL does not require that older applications be manipulated or touched in any way. The data that comes out of the legacy application can be isolated. And data coming from different applications can be integrated. The data is in transit in any case, so this becomes an ideal place in which to examine and audit data, and to make changes if needed.

The third place where data quality can be applied is once the data has arrived in the data warehouse. Over time data changes, and what was accurate and proper one year is not accurate and proper the next. So even if the data is loaded perfectly into the data warehouse, there still is a need for periodic adjustment of data based on changes in business conditions that have occurred over time.

Volumes of Data

The volume of data grows beyond any expectations in a data warehouse. Once terabyte data warehouses were a dream; today they are a reality. In fact, it is not unheard of to build petabyte data warehouses.

As data volumes grow large, the techniques and approaches to their management change. One of the most important characteristics of a data warehouse growing is the appearance of dormant data that just sits there taking up space and costing money. When the data warehouse was small, all or nearly all of the data that resides inside of it was used. But as the data warehouse grows large, increasingly large amounts of data reside in the warehouse in an unused state.

When a data warehouse is around the 100-GB range, there may be only 10 to 20 percent dormant data. But as a data warehouse approaches a terabyte, it is not unusual for the dormancy ratio to go to 50 to 75 percent. And as a data warehouse goes beyond several terabytes, the amount of dormant data frequently approaches 90 to 99 percent.

It is wasteful for the corporation to continue to increase the size of a data warehouse when the proportion of data that is dormant increases as well. In addition, increasing the size of a data warehouse when there is much dormant data grossly hurts performance.

Removing Dormant Data

Dormant data needs to be periodically removed from disk storage and placed on another media. Active data is placed on disk storage and is managed and accessed in a normal manner, while data that is inactive is placed in a physically separate facility, sometimes called alternate storage or near-line storage. This causes the cost of the data warehouse to drop dramatically and the speed with which data can be accessed to increase.

To make the marriage between dormant data and actively used data residing on disk work well, a technology called a *cross-media storage manager* (CMSM) may be utilized. The CMSM sits between disk storage and alternate storage and manages the traffic between the two environments so that the end user is presented with a seamless view of the data residing in the data warehouse.

Meta Data

One of the essential aspects of the data warehouse is meta data. *Meta data* is the information about the contents of what has come to be termed the corporate information factory, or CIF. (The evolution of the CIF is discussed in detail in the next section.) Every application has its own meta data, which is distributed across the entire landscape of architectural components. The meta data has two functions: to describe the data that is found in the architectural component and to exchange meta data with other components.

Meta data in the data warehouse plays several roles. One role is describing what data resides where for normal usage. It also acts as a coordinator between different services from ETL to information access. The different services of the architecture have very different foundations and functions. Some serve under one database management system, others under another database management system. Some services operate under one type of multidimensional technology, and other services operate under other multidimensional technologies. And there is a very different function for all of the services. For the services to operate in unison with each other, there must be coordination from one service to the next. The coordination is achieved through meta data being passed from one architectural layer to another.

There are distinctly different kinds of meta data, including technical meta data, operating meta data, and business meta data. *Technical meta data* describes the structure and content of the different types of data. This type of data has been housed in data dictionaries and repositories for a long time. Operational meta data is the metrics that are generated by the day-to-day operation of the data warehouse. Metrics such as records passed from one software component to another, length of operation of a program, number of records in a database, and so forth make up operating meta data. *Business meta data* is couched in terms the business person understands. Business definitions, business formulae, and business conditions all make up business meta data. All three types of meta data are needed for controlling the operation of a data warehouse.

One concern is the integrity of the meta data. To maintain control and believability of meta data when it is distributed across many different components, a certain protocol for the meta data is necessary. To maintain integrity of meta data across a distributed environment, each unit of meta data must be unique and each unit of meta data must have one owner. The owner of the meta data is the only person or organization that has the right of update, creation, and deletion of the unit of meta data. Everyone else becomes a sharer of the meta data. As meta data is passed from one node to the next, careful track of ownership of the meta data must be kept.

Evolution of Information Processing

For a variety of reasons information was once nearly impossible to get out of applications. Corporate applications were not integrated, applications did not contain historical data, and applications were housed in technology that was not easy to access. The result was frustration for the end user.

This frustration led to the notion of a data warehouse. The data warehouse was a radical departure from the database theory of the day that mandated all data be gathered into a single database. The data warehouse concept focused on different kinds of databases for different purposes. Operational transaction processing was served by one type of database, and informational processing was served by another. The data warehouse called for data to be integrated and stored over time in a physically separate database technology that was optimal for access and analysis of information.

Data marts grew from the data warehouse, DSS applications appeared, and the data warehouses grew in size to a point where the volumes of data in a warehouse easily exceeded the size of earlier databases by several orders of magnitude. Early online databases were considered large at 10 GB. Today, 10-TB data warehouses are considered large, and 10 TB is three orders of magnitude larger than 10 GB. Even with the data warehouse, other forms of informational technology were needed.

As the different architectural structures appeared in conjunction with the data warehouse, even more architectural structures were spawned. Soon there were operational data stores (ODSs), data mining and exploration facilities, alternative forms of storage, and so forth. The data warehouse gave birth to different forms of environments that accomplished very different types of informational processing. At the center of these structures was the data warehouse. The data warehouse provided the granular data that was reshaped into many different forms in order to feed the many different forms of decision support processing. The architectural framework that blossomed was called the *corporate information factory* (CIF). Figure 1.7 shows the progression of the world of informational processing and data warehouse. At the forefront of informational processing is the data warehouse and the larger architecture that centers around the data warehouse—the CIF.

Often the architecture that resulted was called *hub-and-spoke architecture*. Similar to the flight plans and strategies used by the commercial airline industry where a city acts as the hub and enables connection to various destinations through routes, the data warehouse sits at the hub and various analytical applications and data marts act as destinations. The process of delivering information to the destinations is then analogous to the routes or spokes.

The CIF represents the progression of thought and development that occurred with informational processing. This progression occurred in the same timeframe that SAP and Enterprise Resource Planning (ERP) were developing and maturing. It was inevitable that the worlds of SAP/ERP and data warehouse/corporate information factory would merge.

From the standpoint of timing, the CIF was intellectually articulated before SAP BW became available. However, this fact hardly means that the business world went out and built the CIF immediately. Yet that SAP BW followed the intellectual establishment of the CIF in no way diminishes or tarnishes the value of the SAP BW product. Indeed, SAP BW provides an easy path to the actualization of the CIF for many organizations.

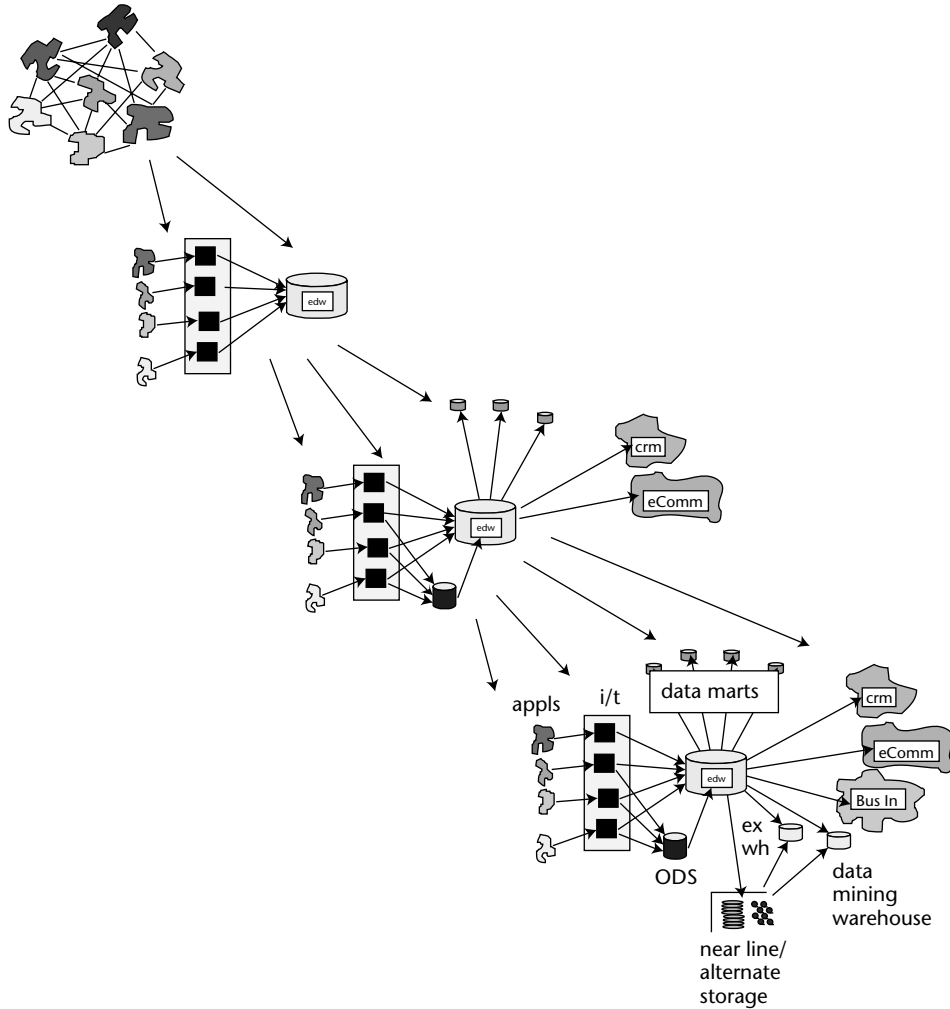


Figure 1.7 The evolution of the CIF.

In many ways the CIF is like a city plan. A city plan takes years and decades to develop. In the case of grand cities built from a plan (such as Washington, DC), it may take decades for the city plan to be realized. And thus it is with the CIF. The CIF is a blueprint and as such may require many years for implementation. The CIF in a robust state is pictured in Figure 1.8.

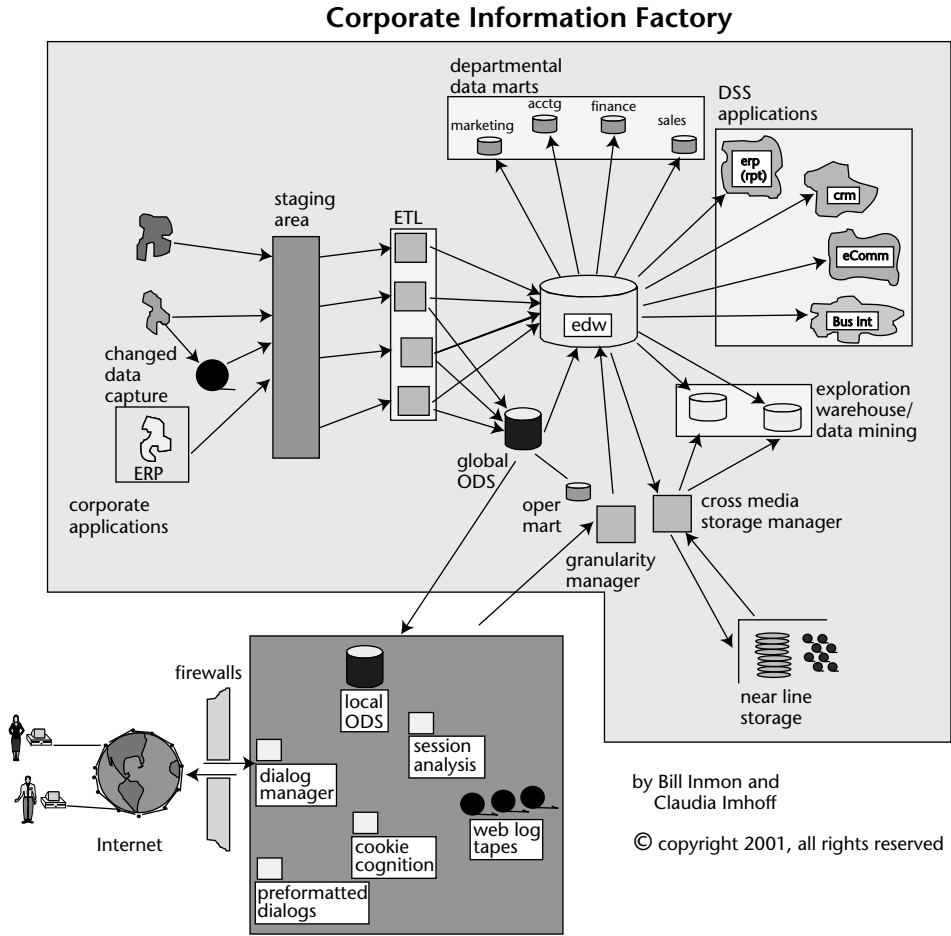


Figure 1.8 The CIF and the Web-based e-business environment.

Setting the Stage for Business Intelligence

Once the data warehouse has been constructed, the stage is set for effective business intelligence. There are many different forms of business intelligence. As shown in Figure 1.9, business intelligence is found in exploration and data mining, data marts, e-business support, and decision support system (DSS).

In a sense, a data warehouse becomes the support infrastructure for business intelligence. Once the data warehouse is built, it becomes very easy and natural to build business intelligence on top of the foundation.

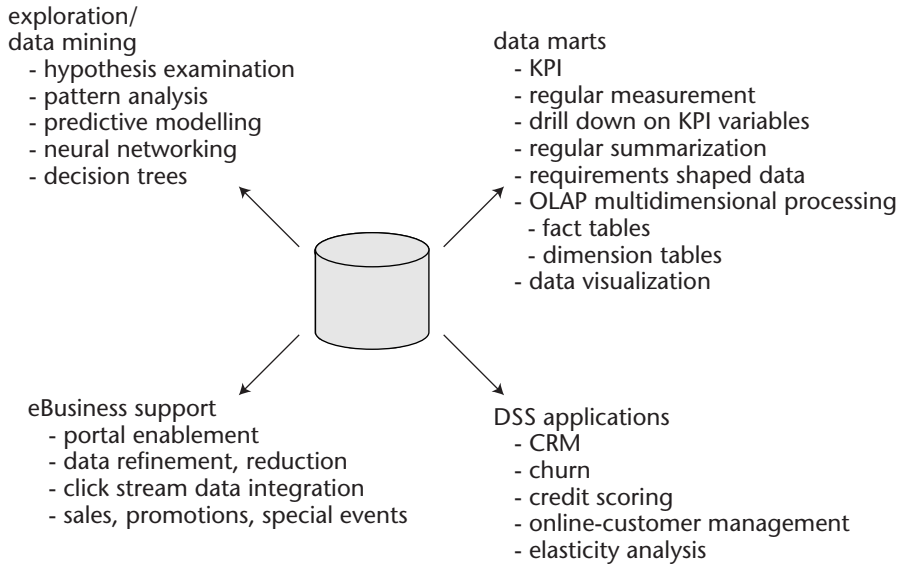


Figure 1.9 Forms of business intelligence.

Summary

In the beginning were simple applications, followed by online applications. Soon a spiderweb of systems and data was created. With the spiderweb came many difficulties, such as redundancy of data, lack of integrity, and the inability to make changes. In parallel it was recognized that there was a fundamental difference between operational data and informational or DSS data, requiring a change in architecture. The spiderweb environment needed to be split into two kinds of processing: operational processing and informational processing. Operational processing was done out of an ERP system like SAP, and informational processing was done out of a data warehouse, which became the “single version of truth” for the corporation at the most granular level.

Data warehouses—whether from SAP environments or non-SAP environments—share some common elements:

Data model. The ERD determines what the major subject areas are. The data item set determines what the attribution and keys will be.

Different physical tables or databases linked by a common key structure. The different tables are built in an iterative manner. It is patently a mistake to build the data warehouse in an all-at-once manner, or a “big bang” approach.

ETL processing. ETL processing accesses legacy/operational data and prepares the data for entry into the data warehouse. ETL processing represents the point of integration for data found in the data warehouse.

Granular data. Granular data can be examined in many ways, and it sits in wait for unknown requirements.

Robust history. Typically five years' worth of data is found in the data warehouse environment.

The data warehouse sets the stage for the many different forms of business intelligence and is the central component in the CIF. In Chapter 2, we discuss how SAP has evolved from a provider of online transactions processing applications to a business intelligence solutions provider.