Chapter 8

Building Data and Document-Driven Decision Support Systems

INTRODUCTION

In recent years, most large companies and many other large organizations have implemented database systems called data warehouses and some have also implemented document management and On-line Analytical Processing (OLAP) systems. Some organizations have implemented Business Intelligence (BI) technologies and some have created Executive Information Systems (EIS). Many managers and information systems specialists are interested in learning more about these relatively new types of data-driven and document-driven Decision Support Systems (DSS). For many years, the prospects and problems of providing managers with real-time management information have been discussed and debated (cf., Dearden, 1966). The debate about costs, advantages, problems, and possibilities must continue. Managers need to retrieve and analyze large structured and unstructured data collections for decision support.

The expanded DSS framework categorizes business intelligence systems, data warehouses, EIS, spatial DSS, and OLAP systems as data-driven DSS. In general, a data-driven DSS is an interactive computer-based system that helps a decision maker use a very large database of business data and, in some systems, data about the external environment of a company. For example, a system may have data on both a company's sales and on its competitors' sales. Some of the data is very detailed transaction data and some is a summary of transactions. In most implementations of data-driven DSS, users of the system can perform unplanned or ad hoc analyses and requests for data. In a data-driven DSS, managers process data to identify facts and draw conclusions about relationships and trends. Data-driven DSS help managers retrieve, display, and analyze historical data.

Document-driven DSS are defined as systems that integrate "a variety of storage and processing technologies to provide complete document retrieval and

analysis." The Web now provides access to large document databases, including databases of hypertext documents, images, sounds, and video. Examples of documents that could be accessed by a document-driven DSS are policies and procedures, product specifications, catalogs, news stories, and corporate historical documents, including minutes of meetings, corporate records, and important correspondence. A search engine is a powerful decision-aiding tool associated with a document-driven DSS (cf., Fedorowicz, 1993, pp. 125–136; Swanson and Culnan, 1978; Power, 2001). Knowledge management, Web, and DSS technologies are used to build document-driven DSS.

Data and document-driven DSS are often very expensive to develop and implement in organizations. Despite the large resource commitments that are required, many companies have implemented these types of DSS. Technologies are changing and managers and MIS staff will need to make continuing investments in these categories of DSS software. So, it is important that managers understand the various terms and systems that use large databases to support management decision making. This chapter emphasizes: comparing data and document-driven DSS; identifying subcategories of data-driven DSS, comparing structured DSS data and operating data, understanding an interconnected data-driven DSS architecture, implementing data and document-driven DSS, and finding success in building DSS with large structured and unstructured databases. Now, let's begin our exploration of these two general categories of DSS by discussing the differences and similarities between them.

COMPARING DATA AND DOCUMENT-DRIVEN DSS

Document-driven DSS is a relatively new category of decision support. There are certainly similarities to the more familiar data-driven DSS, but there are also major differences. Document-driven DSS help managers process "soft" or qualitative information, and data-driven DSS help managers process "hard" or numeric data. Both categories of DSS come in various shapes and sizes. Some systems support senior managers and others support functional decision makers on narrowly defined tasks. The Web has increased the need for, and the possibilities associated with, document-driven DSS.

A defining difference between the two categories of DSS is that data-driven DSS help managers analyze, display and manipulate large structured data sets that contain numeric and short character strings while document-driven DSS analyze, display, and manipulate text including logical units of text, called documents (cf., Sullivan, 2001).

Another defining difference is the analysis tools used for decision support. Data-driven DSS use quantitative and statistical tools for ordering, summarizing, and evaluating the specific contents of a subject-oriented data warehouse. Document-driven DSS use natural language and statistical tools for extracting, categorizing, indexing, and summarizing subject-oriented document warehouses.

What are the similarities? First, both systems use databases with very large collections of information to drive or create decision support capabilities. Second, both types of systems require the definition of metadata and the

cleaning, extraction, and loading of data into an appropriate data management system using an organizing framework or model.

Third, building either type of system involves understanding the decision support and information needs of the targeted users. Also, because user needs are hard to anticipate the tendency is to store large amounts of data or documents that may not be immediately needed. Rapid application development or prototyping is sometimes possible for small scale systems, but a more structured SDLC approach is needed for enterprise-wide data or document-driven DSS. Neither type of system can meet all of the decision support needs of all managers in an organization. The best approach is to try to meet a specific, well-defined need initially and then incrementally expand the structured data or documents that are captured and organized in the foundation data/document management system.

DATA-DRIVEN DSS SUBCATEGORIES

The broad category of data-driven DSS generally includes tools to help users "drill down" for more detailed information, "drill up" to see a broader, more summarized view, and "slice and dice" to change the data dimensions they are viewing. The results of "drilling" and "slicing and dicing" are presented in tables and charts. There are four main subcategories of data-driven DSS; data warehouses, OLAP systems with multidimensional databases, Executive Information Systems (EIS), and spatial DSS.

Data Warehouses

A data warehouse is a specific database designed and populated to provide decision support in an organization (cf., Gray and Watson, 1998). It is batch-updated and structured for rapid on-line queries and managerial summaries. Data warehouses contain large amounts of data—500 megabytes and more. According to data warehousing pioneer Bill Inmon (1995), "A data warehouse is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management's decision making process."

What does Inmon mean by his four characteristics of a data warehouse? *Subject-oriented* means it focuses on subjects related to business or organizational activity like customers, employees and suppliers. *Integrated* means the data from various databases is stored in a consistent format through use of naming conventions, domain constraints, physical attributes, and measurements. *Time-variant* refers to associating data with specific points in time. Finally, *nonvolatile* means the data does not change once it is in the warehouse and stored for decision support. Ralph Kimball (1996), another data warehousing pioneer, states that "a data warehouse is a copy of transaction data specifically structured for query and analysis."

A related term is a "data mart." A data mart is a more focused or a single-subject data warehouse. For example, some companies build a customer data mart rather than a multi-subject data warehouse. Such a focused data mart would have all of the business information about a company's customers. Many

organizations and businesses are starting their enterprise-wide data warehouses by building a series of focused data marts. Data warehouses and data marts are often accessed using ad-hoc query or report and query tools. Some authors have combined data warehousing and OLAP. The two terms should be recognized as different subcategories of data-driven DSS.

On-Line Analytical Processing (OLAP)

OLAP and multidimensional analysis refers to software for manipulating multidimensional data. Even though one can have multidimensional data in a data warehouse, OLAP software can create various views and more dimensional representations of the data. According to Nigel Pendse at the OLAPReport.com, OLAP software provides fast, consistent, interactive access to shared, multidimensional information. Pendse calls these characteristics the FASMI test, an acronym for fast analysis of shared, multidimensional information test. What does the FASMI test mean?

FAST means that the system delivers most responses to users within about five seconds. ANALYSIS means that the system can cope with any business logic and statistical analysis that is relevant for the application and the user. SHARED means that the software has security capabilities needed for sharing data among users. MULTIDIMENSIONAL is an essential requirement. An OLAP system must provide a multidimensional, conceptual view of the data. INFORMATION means the software can support all of the data and derived information that managers need.

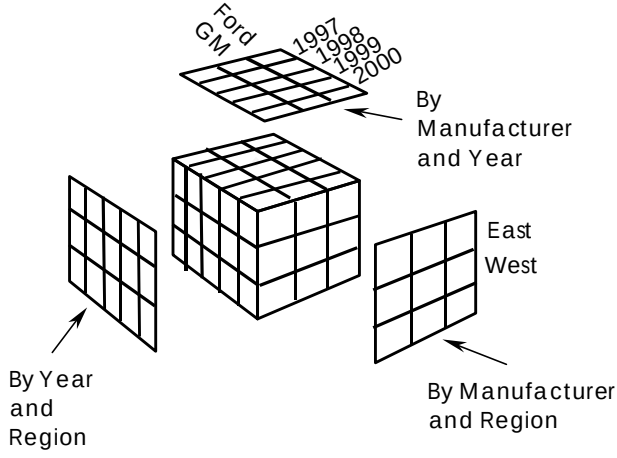

Figure 8.1 An Example of a Multidimensional Data Cube.

OLAP software usually accesses a multidimensional database**.** A multidimensional database captures and presents data as a multidimensional array or data cube. Variables hold data in the database. The multidimensional

database management system creates arrays of values, usually numeric, that are "dimensioned" by relevant attributes. For example, the attributes "year," "manufacturer," and "region" are dimensions of a "units sold" variable. This three-dimensional array can be visualized as a cube of data (see Figure 8.1). Arrays with more dimensions are often created, but such arrays are harder to visualize.

Multidimensional databases can have multiple variables with a common or a unique set of dimensions. A multidimensional view of data is especially powerful for OLAP applications. For example, one can sum units in dimensions. A relational database software package can also be used to structure data to support rapid, multi-dimensional queries. A Star schema is a typical structure implemented for multidimensional data using a relational database management system (cf., Gray and Watson, 1998). A Star schema has a central table of facts, often called a Fact Table, and dimension tables linked to it by foreign keys like StoreID or ProductID (see Figure 8.2). The star is a picture of the way the data is being stored. The basic factual information is in the middle of the star. This type of application, where multidimensional data is stored in a relational database management system has been called ROLAP, short for Relational OLAP.

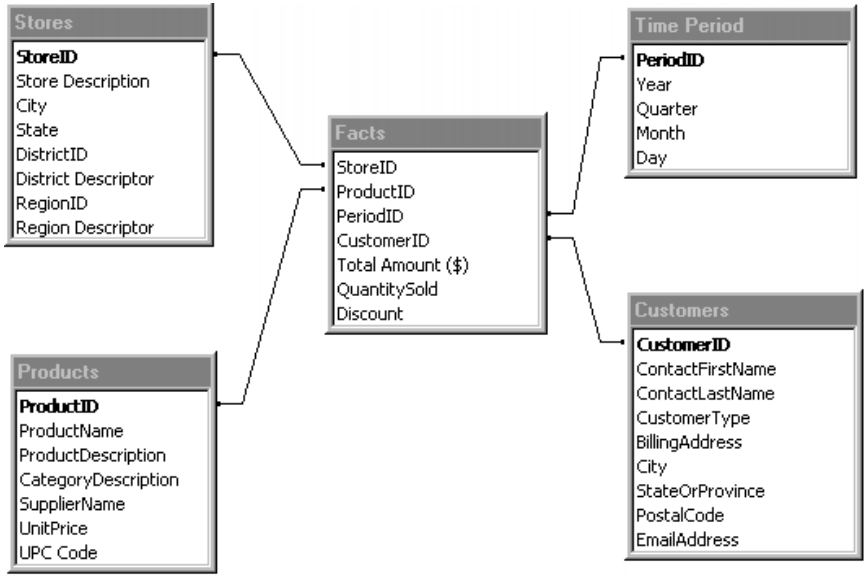

Figure 8.2 Star Schema Diagram

Like other data-driven DSS, OLAP usually provides drill-down and drill-up capabilities. Software reviewer Jay Tyo (1996) divided OLAP tools into five broad types that he labeled: stand-alone desktop OLAP tools; integrated desktop tools; relational OLAP tools; personal multidimensional databases; and other OLAP tools. The variety of products has only expanded in recent years with the

introduction of Web-based OLAP. Categorizing data-driven DSS products is complex and difficult. New products are continually being introduced. So, managers and DSS analysts are confronted with a wide array of different types of OLAP and Business Intelligence products. The OLAP vendors have created a number of technical terms that can be somewhat confusing; for help with OLAP and data-driven DSS terms, check the *Guide to OLAP Terminology* created by the OLAP Council and the DSS Glossary at DSSResources.COM.

Business Intelligence (BI) is sometimes used interchangeably with OLAP and other synonyms for DSS. BI is a popularized umbrella term introduced by Howard Dresner of the Gartner Group in 1989 to describe a set of concepts and methods to improve business decision making by using fact-based support systems. A Business Intelligence system is a data-driven DSS. The most common "business intelligence" software is for querying a database and creating a report.

Executive Information Systems

Executive Information Systems (EIS) are computerized systems intended to provide current and appropriate information to support executive decision making for managers (cf., Watson, Rainer, and Houdeshel, 1992). The emphasis is on graphical displays and an easy-to-use interface that present information from the corporate database. EIS are designed to provide canned reports or briefing books to top-level executives. They offer strong reporting and drill-down capabilities.

EIS differ from traditional information systems in a number of ways (Kelly, 1997):

1. EIS are specifically tailored to an executive's information needs.
2. EIS are able to access data about specific issues and problems as well as aggregate reports.
3. EIS provide extensive on-line analysis tools including trend analysis, exception reporting, pivot tables, and "drill-down" capability.
4. EIS access a broad range of internal and external data.

Differentiating EIS from other sub-categories of data-driven DSS has historical value in that such systems were developed separately from model-driven DSS and report and query tools, but it also helps analysts understand that senior manager's have different decision support needs. EIS are intended to help senior executives find problems, identify opportunities, identify trends, and make fact-based decisions. EIS usually let managers pivot and change dimensions and "drill down" for more information in structured information displays. EIS are designed to avoid data overload for busy managers.

EIS and data-warehousing technologies are converging in the marketplace. EIS used proprietary databases that required many staff people to update, maintain, and create. This was very expensive. In addition, the data became obsolete quickly. New Business Intelligence and OLAP systems require much

less staff support. Data warehouse and OLAP technologies have made EIS more powerful and more practical.

EIS report key business results to managers. The performance measures in the EIS must be easy to understand and collect. Whenever possible, data should be collected as part of routine work processes. An EIS should not add substantially to the workload of managers or staff. Balanced Scorecard measurement software can be used in some firms to expand the measures used in an EIS (check balancedscorecard.org).

In general, EIS are enterprise-wide, data-driven DSS that help senior managers analyze, compare, and highlight trends in important variables so that they can monitor performance and identify opportunities and problems. EIS increase the ability of senior executives to monitor many activities and may help in reducing the number of management levels in an organization.

Geographic Information Systems and Spatial DSS

The final subcategory of data-driven DSS that should be recognized is a spatial DSS (cf., Crossland, Wynne and Perkins, 1995) built using Geographic Information Systems (GIS) technologies. A GIS is a support system that represents data by using maps. Spatial DSS help a manager access, display, and analyze data that have geographic content and meaning. This type of system has been available for many years (cf., Sprague and Carlson, 1982). Some examples of spatial DSS include systems for crime analysis and mapping, customer demographic analyses, and political voting patterns analysis.

Spatial DSS applications are common in routing and location analysis, marketing, and traditional application areas of GIS in disciplines such as geology, forestry, and land planning (cf., Keenan, 1997).

GIS software provides a development environment for spatial DSS. Even limited functionality GIS software provides the ability to zoom in on a map and to display or highlight different data. A GIS provides database support that is designed to allow for the effective storage of spatial data. Also, GIS software provides a link between the user interface and database so a user can query and analyze the spatial data.

The developments and improvements in GIS software since 1990 make it practical to use off-the-shelf software to build a spatial DSS (cf., Keenan, 1997). An example of this type of software is the ArcInfo8 enterprise GIS software from ESRI (www.esri.com). ArcInfo is intended to help users view and query spatial data. Another widely used desktop mapping product is MapInfo (www.mapinfo.com). Also check Peter Keenan's excellent Web Resource on Spatial DSS at URL mis.ucd.ie/iswsdss/. Another major related Web site is Geographic Information Systems Resources and Links maintained by the U.S. Geological Survey at URL info.er.usgs.gov/research/gis/title.html.

Data-driven DSS have captured the imagination of managers because they can provide much easier access to a vast amount of business data. In a world of speeded-up competition, rapid changes in markets and products, and increased electronic communication, managers want to find their own answers to business questions. Managers are *not* willing or able to wait while financial or marketing

analysts create special reports from databases. Managers are the customers and advocates for data-driven DSS. Because data is the driver of such systems it is important to identify and organize decision relevant data or what might be called DSS data. Matching decision situations and DSS data is the key to building data-driven DSS and making better fact-based managerial decisions. Now let's compare and contrast DSS data and operating data.

COMPARING DSS DATA AND OPERATING DATA

First, it is important to remember that operating data and DSS data serve different purposes. In general, DSS data is data about transactions and business occurrences; operating data is a detailed record of a company's daily business transactions. DSS data is created to provide tactical and strategic business meaning to operating data and relevant external data. The difference in purpose means that the data formats and structures will likely differ. Managers and DSS analysts must recognize that DSS data and operating data differ in terms of six major factors: the data structures, the time span, the summarization of data, data volatility, data dimensions, and metadata (cf., Rob and Coronell, 1997). Table 8.1 is a summary of the factors and the differences between operating and DSS data. The next six sections examine these differences in more detail.

Factors	Operating Data	DSS Data
Data Structures	normalized	integrated
Time Span	current	historical
Summarization	none	extensive in some systems
Data Volatility	volatile	non-volatile
Data Dimensions	one dimension	multiple dimensions
Metadata	desirable	required and important

Table 8.1. Comparing Operating and DSS Data.

Data Structures

What are the differences between operating and DSS data structures? It is useful to examine the extent and nature of the differences in format and structure. Often, operating data are stored in a relational database management system. These relational transaction systems have data structures called tables that have been highly normalized. The tables are normalized to avoid anomalies in the data when transactions like updating, adding records, and deleting records occur. Normalization is the process of reducing a complex data structure into its simplest, most stable, structure. The process involves removing redundant attributes, keys, and relationships from a conceptual data model.

In general, in an operating data storage or transaction system, both the software and the hardware are optimized to support transactions for the daily

operations of a company. For example, each time an item is sold, it must be recorded and accounted for in appropriate transaction tables. Also, related data like customer data and inventory data are updated in transaction processing systems. In order to provide effective and efficient update performance, transaction systems store data in many small tables, each with a minimum number of fields. Thus, a product purchase transaction might need to have data elements recorded in five or more different tables. For example, records may be added or updated in an invoice table, an invoice line table, a discount table, a store table, and a department table.

Although this structural approach of creating many small tables is effective in a transaction database, it is not appropriate for DSS data. Queries will tend to be slow, and, in many cases, tables will need to be joined to complete a query. For example, to create an invoice from an operating database to mail to a customer, all of the tables may need to be joined. In a large database of transactions, joining tables is time-consuming and uses extensive computing system resources. Operating data are usually stored in many tables and the stored data relates to a specific transaction. DSS data are generally stored in many fewer tables. DSS data does not always include the details of each operating transaction but does include transaction summaries. In general, DSS data are integrated from multiple operating databases, and are sometimes aggregated and summarized in the database to support predefined decision support needs. Also, DSS data may have data redundancies in the data structures if that will speed up queries.

The different data components of data warehouses often include: metadata, current detail data, older detail data, lightly summarized data, and highly summarized data. Extensive normalization is not appropriate for DSS data, and some normalization will actually reduce the processing efficiency of a data-driven DSS. Normalization is not needed because the data will not be changed once it is in the database, and hence, anomalies or errors from updating will not occur.

Time Span

Operating data shows the current status of business transactions. DSS data are a snapshot of the operating data at given points in time. Therefore, DSS data are an historic time series of operating data. In a DSS data store, operating data are stored in multiple "time slices." Inmon (1993) says the DSS data is "time-variant." This characteristic is analogous to putting a time stamp on DSS data when it is loaded in the database or data store.

Summarization

DSS data can be summarized in the DSS data store, and disaggregated data can be summarized by analytical processing software. Data can be brought from a DSS database into a multidimensional data cube to speed up analysis. Some DSS databases consist exclusively of summarized—or what is often called derived—data. For example, rather than storing each of 10,000 sales

transactions for a given retail store on a given day, a DSS database may contain the total number of units sold and the total sales dollars generated during that day. DSS data might be collected to monitor total dollar sales for each store or unit sales for each type of product. The purpose of the summaries is to establish and evaluate sales trends or product sales comparisons that will serve decision needs. Managers may want to ask questions like: What are sales trends for Product X? Should Product X be discontinued? Has advertising been effective as measured by sales changes? All of these questions can be answered using summarized data. Operating data is not summarized within a transaction database.

Data Volatility

Only two kinds of operations occur in a data warehouse or DSS database: loading of data and accessing data. Data can be added in batches but there is no on-line updating and changing of data. So the DSS data is non-volatile. Once it is loaded it does not change. Operating data, on the other hand, is volatile. Operating data changes when a new transaction occurs. The database management system for a transaction system records new transactions and changes in transactions.

Data Dimensions

Having multiple dimensions is probably the most distinguishing characteristic of DSS data. From a manager's and a DSS analyst's point of view, DSS data are always related in many different ways. For example, when managers analyze product sales to a specific customer during a given span of time, they are likely to ask multidimensional questions. A manager may ask, "How many products of type X were sold to customer Y during the most recent six months?" DSS data can be examined from multiple dimensions, for example, product, region, and year. The ability to analyze, extract, and present data in meaningful ways is one of the major differences between a data-driven DSS and a transaction processing system. In contrast to DSS data, operating data has only one dimension.

Metadata

In a data-driven DSS it is important to develop and maintain metadata about the DSS data. Metadata is defined as "data about the data" in a DSS database. Data dictionaries are often created for transaction systems, but because DSS data may come from many sources, creating a new dictionary and metadata is especially important for a data-driven DSS. Also, the database designer must integrate DSS data that comes from different sources. The data dictionary provides a reference about how data has been combined from various data sources.

Metadata provides a directory to help the Database Management System for the data-driven DSS locate the contents of a data warehouse or data store.

Metadata is a guide to mapping data as it is transformed from the operating environment to the data warehouse environment, and it serves as a guide to the algorithms used for summarization of current detailed data. Metadata is semantic information associated with a given data element. Semantic information explains the meaning of what is recorded and stored in a DSS data store.

Metadata must include business definitions of the data and accurate, understandable descriptions of data types, potential values, the original source system, data formats, and other characteristics. Metadata also includes the names of variables, length of fields, valid values, and descriptions of data elements. Metadata protects a data warehouse or database from changes in the schema or design of source systems.

Data-driven DSS must have high quality data; inaccurate data can result in incorrect or poor decisions. High quality data is accurate, timely, meaningful, and complete. Assessing or measuring the quality of data sources is a preliminary task associated with evaluating the feasibility of a data-driven DSS project.

The above comparison of DSS data and operating data suggests some architectural issues related to building a data-driven DSS. The next section addresses some data-driven DSS software architecture issues more systematically.

AN INTERCONNECTED DATA-DRIVEN DSS ARCHITECTURE

DSS designers should begin building a new data-driven DSS by researching other data-driven DSS to identify a data model and an appropriate DSS architecture. The overall goal should be to understand a typical data-driven DSS's components and interfaces, how it fits into the typical organization, and what the typical reasons are for success or failure. In some cases vendors have developed data models and designs for specific industries and applications that can serve as a guide. After understanding data-driven DSS architectures in general, developers should map a template onto their company's specific situation. Developers need to determine the subjects that will be included and the questions that may be asked by decision makers.

A useful starting point is to examine the components in a data-driven DSS software architecture. At a minimum, DSS designers need to provide data structures for a data store, guidelines for a data extraction and filtering management tool, interfaces for a query tool, and some predefined charts and tables for use with a data analysis and presentation tool. The following paragraphs examine these four interconnected software architecture components.

The data store component consists of one or more databases, built using a relational database management system, a multidimensional database management system, or both types of systems. As noted, business data is extracted from operating databases and from external data sources. The external data sources provide data that cannot be found in company transaction systems but that are relevant to the business, such as stock prices and market indicators. The data store is a compilation of many "snapshots" of a company's financial,

operating, and business situation. When developers create DSS data for the data store, they summarize and arrange the operating data in structures that are optimized for analysis and rapid retrieval of data. The "aging process" developed for a data store moves current detail data to older detail data based on when the data was loaded. This archiving occurs each time a batch update is performed. In most situations only summarized data is indexed in the data store.

The data extraction and filtering component is used to extract and validate the data taken from the operational databases and the external data sources. For example, to determine the relative market share by selected product line, the DSS requires data about competitors' products. Such data may be located in external databases provided by industry groups or by companies that market such data. As the name implies, this component extracts the data from various sources, filters the extracted data to select the relevant records, and formats the data so it can be added to the DSS data store component.

A decision support analyst or a manager can create the queries that access the DSS database using a report and query tool. Developers usually customize the query tool interface for managers so it is easier to use. The query tool actually accesses the data store and retrieves requested data.

Finally, an end user analysis and presentation tool helps a manager perform calculations and select the most appropriate presentation format. For example, managers may want to display data using a pivot table summary report, a map, or a bar chart. The query tool and the presentation tool are often the "front end" of a data-driven DSS. Client/server and Web technologies enable these end-user components to interact with the other components to form a complete data-driven DSS software architecture.

Once the software architecture is developed for a specific DSS, designed for a specific company and a specific purpose, a DSS development team still faces many challenges associated with implementing a new data-driven DSS.

IMPLEMENTING A DATA-DRIVEN DSS

Organization-wide Information System (IS) development projects like data-driven DSS are subject to numerous constraints. Some of these constraints are based on available funding. A large data warehouse can cost USD $ 2 to 3 million for software, hardware, staff development time, and training costs and can take two to three years to build. Other constraints are a function of management's view of the role played by an IS department and of management information and DSS requirements. Also, constraints may be imposed by corporate culture conflicts. It is important to identify issues that must be confronted when implementing a data warehouse, OLAP system or other data-driven DSS.

It is very important to remember that a DSS data store is not a static database. Instead, it will be supplemented regularly with new data. Because the data store is a foundation of a modern data-driven DSS, the design and implementation of a sophisticated data store provides an infrastructure for company-wide decision support. The decision support infrastructure also includes hardware, software, people, and procedures. A data store is a critical

component, but it is not the only important component. The structure of the data store and its implementation must be examined in the context of the entire DSS infrastructure.

The technical aspects of creating a new database must be addressed. A new data-driven DSS must provide required analysis capabilities with acceptable query performance, and the DSS must support the data analysis needs of decision makers.

Traditional database design procedures must be adapted to fit the requirements of building a large DSS data store. Data is derived from transaction databases, so a DSS analyst must understand the transaction database designs. It is difficult to produce good DSS data when transaction databases are of poor quality or are inaccurate. So how should a data warehouse or data-driven DSS be developed?

A General Data-Driven DSS Development Process

Various consultants have customized their data-driven DSS development processes. Chapter 4 discussed the two general approaches called Systems Development Life Cycle and Rapid Prototyping. For small projects like a data mart, one can use Rapid Prototyping. For large projects, the following steps, based on a typical data warehouse development process (see Rob and Coronell, 1997), are appropriate. This decision-oriented design and development process includes five steps (see Figure 8.3):

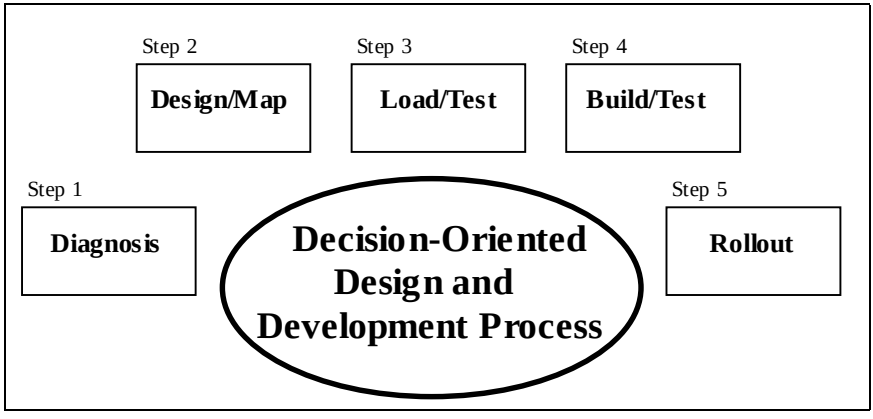

Figure 8.3 Decision-Oriented Process for a Data-Driven DSS.

The first step is *Initial Data Gathering or Diagnosis*. This step involves identifying and interviewing key future DSS users, defining the main subjects of the DSS, identifying the transaction data model, defining ownership of data, assessing frequency of use and updates, defining end user interface requirements and defining any outputs and representations. The emphasis on decision makers and decisions should be maintained in subsequent steps.

The second step is *Designing and Mapping the Data Store*. In a relational DBMS environment, the first step is to design the Star Schema and identify facts, dimensions, and attributes. Then, one creates Star Schema diagrams, attribute hierarchies and aggregation levels. These conceptual models then need to be mapped to relational tables. In a multidimensional database environment, the key variables and dimensions need to be defined. The data store houses the relevant DSS data.

The third step is *Loading and Testing Data*. Creating the DSS database involves preparing to load data, defining initial data to load, and defining update processes. Then analysts define transformations of the transaction and any external data, map from the operational transaction data, integrate and transform the data. Next, analysts load, index and validate the data, and finally verify metadata and data cubes or Star Schemas.

The fourth step is *Building and Testing the Data-driven DSS*. Analysts need to create menus, develop output formats, build anticipated queries, test interfaces and results, optimize for speed and accuracy, engage in end user prototyping and testing, and provide end user training in a development environment. Decision makers need to be heavily involved in building and testing the new data-driven DSS.

The final step is *Rollout and Feedback*. This step involves actually deploying the DSS, providing additional training, getting user feedback, maintaining the system, and in many cases expanding and improving the DSS. One expects that the new DSS improves decision making and benefits the company and decision makers.

The above five-step development process needs to be altered in some important ways when one is building an Executive Information System.

Developing an Executive Information System

Information needs of executives change rapidly, so many Executive Information Systems are developed using rapid prototyping tools. Usually, a staff group creates screens and information displays for use in the EIS. The group needs to experiment with how data is presented and receive feedback from users. Determining executive information requirements can be an especially challenging task. Some of the systematic methods like structured interviews may need to be used to supplement reviews of prototype screens.

Although data-driven DSS using query and reporting tools are sometimes developed by end-users as desktop systems, EIS are traditionally more elaborate networked systems developed by IS professionals in cooperation with financial and staff professionals. Determining the critical success factors for an organization can help analysts determine what information should be presented in the EIS. Critical success factors are variables like earnings per share, market share, productivity, or units delivered that influence performance and success for a firm (cf., Rockart, 1979).

If a company wants to update its EIS or create a new capability, a small project team should be organized. According to Kelly (1997), a project leader should organize and direct the project. An executive sponsor or project

champion is needed to promote the project in the organization and review project progress regularly. A technical leader participates in gathering requirements, reviewing plans, and ensuring technical feasibility of all proposals during EIS requirements definition. As the focus of the project becomes more technical, the EIS project team should be expanded to include additional technical staff who will be directly involved in extracting data from legacy systems and constructing the EIS data repository and user interface. An EIS project is similar to data warehouse projects with additional emphasis placed on the design of the user interface.

A well-known example of an EIS is the Lockheed-Georgia Management Information and Decision Support (MIDS) system (cf., Watson, Rainer and Houdeshel, 1992). MIDS was upgraded to a commercial system from Comshare in the early 1990s. A more recent example is an EIS deployed at Pizzeria Uno (www.pizzeriauno.com). In an attempt to boost its competitiveness and profitability, Pizzeria Uno's management requested an EIS that would give executives and field management access to timely information on sales and labor costs. The company installed OLAP server software to support the EIS. Marketing, operations, and finance executives at Pizzeria Uno headquarters access the server's database from their PCs. Regional managers dial into the server each day to download up-to-date data onto their laptop computers. Executives can drill through data hierarchies, manipulate data, view data from different dimensions, such as deep-dish pizza take-home sales versus retail sales, and create reports tailored to their specific informational needs (based on a case at the Pilot Software Web site).

FINDING SUCCESS

Data or document-driven DSS development is usually a company-wide effort and requires many resources, including people and technologies. Building an effective enterprise-wide DSS is usually much harder than implementing a communications-driven DSS or developing a model-driven DSS using rapid prototyping. First, providing company-wide decision support requires a sophisticated information technology architecture. Creating such an architecture requires a mix of people skills, technologies, and managerial procedures that is often difficult to find and implement. For example, storing a large quantity of decision support data is likely to require purchasing the latest hardware and software. Most companies need to purchase high-end servers with multiple processors, advanced database systems, and very large capacity storage units. Some companies need to expand and improve their network infrastructures.

For a data-driven DSS, MIS staff need to develop detailed procedures to manage the flow of data from transaction databases to the data store. Data flow control includes data extraction, validation, and integration. A successful implementation of an enterprise-wide DSS architecture requires the support of people with advanced database design and data management skills.

How can managers increase the chances of completing a successful data- or document-driven DSS project? A number of authors have suggested some lessons they have learned from implementing their data warehouse, document

warehouse and DSS projects. After evaluating the suggestions, the following recommendations seem especially useful for completing a successful data- or document-driven DSS project.

The first recommendation that makes sense is to *identify an influential project sponsor or champion.* The project champion must be a senior manager. For a high-cost, highly visible data or document-driven DSS project, the champion can deal with political issues and help insure that all involved realize they are part of a DSS team. All managers need to stay focused on a company's decision support development goals.

Second, managers should *be prepared for technology shortfalls*. Technology problems are inevitable with enterprise-wide DSS projects. Many times, the technology to accomplish some of the desired DSS tasks is not currently available or is not easily implemented. Unforeseen problems and frustrations will occur. Building any type of DSS requires patience and perseverance.

A third recommendation is to *tell everyone as much as possible about the costs* of creating and using the proposed data or document-driven DSS. Managers need to know how much it costs to develop, access, and analyze DSS data and documents. Remember: These systems can be very expensive.

Next, be sure to *invest in training.* Set aside adequate resources, both time and money, so users can learn to access and manipulate the data or documents in the new DSS. From the start, with a data-driven DSS, get users in the habit of "testing" complex questions or queries. With document-driven DSS, users need to learn how to conduct advanced searches.

Finally**,** *market and promote* the new data or document-driven DSS to the managers who are the intended users of the system. Both types of systems are somewhat novel and manager "buy-in" is important. A data or document-driven DSS is not usually tightly integrated with a single business process so market a new system widely to all who might benefit from access to its capabilities.

CONCLUSIONS AND COMMENTARY

Many different terms are used for the systems labeled in this chapter as data-driven and document-driven DSS, and that is OK. What is important is understanding the concepts associated with helping people access, analyze, and understand large amounts of complex data or documents. Some vendors advertise business intelligence (BI) software, data warehouse systems, multidimensional analysis software, OLAP or EIS. Some people would say that rather than document-driven DSS, the term should be "knowledge management" or "document management" or "knowledge management" systems.

What is fundamental is recognizing that different technologies are needed for systems built using structured data versus documents. Data-driven DSS have evolved from simple verification of facts to analysis of the data and now to sophisticated analysis of very large historical data sets. Data-driven DSS software products are also evolving to better support this wide range of activities. Managers can verify facts using intuitive, easy-to-use query and reporting tools. Decision makers can conduct analyses using OLAP and

statistical tools. Also, managers and DSS analysts can discover new relationships in specific data sets, using knowledge discovery and data-mining tools.

Data and document-driven DSS are evolving in terms of technology and architecture. In recent years, systems have been delivering these decision support capabilities using Web technologies to managers located almost anywhere in the world. The Web is an exciting frontier that is broadening the use of data and documents in decision making. Web-based DSS will be discussed further in Chapter 11.

Data and document-driven DSS are important systems for providing managers with the information they want, when they want it, and in a format that is "decision impelling." Despite these benefits, many companies have difficulties developing, implementing, and maintaining enterprise-wide DSS at an acceptable cost. For large-scope decision support projects, cost control largely depends on the management and political skills of the project manager, the restraint of targeted user managers, and the technical skills of the MIS staff on the project team. Once the diagnosis and needs analysis tasks are completed for a project, future DSS users need to restrain their desire for more information and resist changing the requirements for the data- or document-driven DSS.

When building data-driven DSS, relevant data are organized and summarized in multiple dimensions for fast retrieval and ad hoc analysis. With a document-driven DSS, documents and unstructured data are organized and summarized for fast retrieval and ad hoc analysis. The primary goal of these systems is to help managers transform data and documents into information and knowledge. Both management control and strategic planning activities can be supported by such systems.