Building the Framework for Enterprise Data Analysis in the Semiconductor Industry

John T. Stokes, JJT Inc., Austin, Texas
Roger Thompson, JJT Inc., Dallas Texas

Introduction

Leading semiconductor companies have always recognized the benefit a common framework for engineering data analysis could bring to the bottom line. Because commercial software solutions were not available, manufacturers developed in-house systems to address their unique data analysis problems. Unfortunately, attempts by many companies to implement an enterprise engineering data analysis system have been met with mixed results. Over time, these applications became a resource burden to maintain, and quickly became obsolete due to the pace at which new technology is adopted in the supporting hardware and software infrastructure. This paper will address the importance of building an integrated infrastructure for enterprise data analysis and selecting a data model that provides the optimal balance of data accessibility, efficiency and maintainability.

Need for Data Analysis

Companies who wish to remain competitive in the semiconductor industry are driven by aggressive goals to be the first to market with high quality, low cost products. Currently, the semiconductor industry experiences price per unit erosion of 20% or more annually. There is no indication that this trend is going to change in the near term. The market will continue to expect smaller, faster, cheaper electronics. Consequently, in order for a semiconductor manufacturer to remain profitable and competitive, they must be able to reduce cost per component accordingly. Simply cutting staff or capital expenditures on a one-time basis will not effect the kind of repeated annual cost reductions needed. To reduce overall cost and remain profitable, manufacturers must improve yield, increase productivity of both people and equipment, and expedite time to market of new products with leading edge technology.

Because of inherent fluctuations in any manufacturing process, significant variations occur in components manufactured under the same conditions, which result in lower yields. Yield is the ratio of the number of components that meet the desired specifications with respect to the total number manufactured. Consequently, improving yield is an important objective for manufacturers and an essential factor in reducing costs. Yield improvement is most dependent on how quickly manufacturers can move through the learning curve associated with new process technology. As manufacturers climb the yield curve, production is ramped up by increasing the volume of overall wafers manufactured to maximize factory output capability.

Manufacturers have traditionally focused their efforts for shaving cost and increasing throughput in the area of production equipment and process technologies. However, many opportunities exist for reducing cost by improving manufacturing efficiencies. The inability to quickly analyze data to determine the root cause of yield loss directly impacts manufacturing costs. Excessive waste of materials, low resource productivity, and under utilized equipment contributes to driving up costs.

Because most companies lack the basic infrastructure for data analysis, the process of accessing and combining data for analysis can take days and sometimes weeks. In most cases, the investment has already been made to collect and store data from manufacturing and test systems. However, the organization of the data within the different databases is not conducive for data analysis and problem solving. As a result, the benefits of collecting and storing the data is never fully realized. The problem escalates over time as more data is collected and the data chaos is perpetuated.

As manufacturers enter the next cycle of manufacturing and process technology, information driven manufacturing will be a requirement to remain competitive. The infrastructure for informed manufacturing is designed to efficiently provide a platform for data analysis to answer questions regarding complex manufacturing processes. In addition, data mining techniques will provide answers to questions we did not know to ask. The information infrastructure is designed to facilitate learning, disseminate information, and promote knowledge sharing throughout the organization.

Problems with Data Analysis in the Semiconductor Industry

In the semiconductor industry, the sheer volume of data, which must be accessed, managed, and manipulated before any analysis can occur, is often viewed as the most significant problem with data analysis. A tremendous amount of data is collected from various equipment and manufacturing systems and stored in multiple databases scattered throughout the organization. From a decision support standpoint, these data assets are an untapped source of information for reducing manufacturing costs, accelerating through the learning curve, shortening cycle times, improving quality, and bringing new products to market faster. Because of the abundance of data, many unexplored opportunities exist where the appropriate data analysis could bring extremely rewarding results to manufacturers.

As the semiconductor industry moves into the next technology cycle, the problems associated with engineering data analysis will be compounded unless the right solution is in place. The amount of data currently collected in manufacturing and test databases will pale in comparison to the amount collected in the near future. By moving from 200mm(8 inch) to 300mm(12 inch) wafers, manufacturers can increase the number of die on a single wafer by 2.5 times that of a 200mm wafer. In addition, devices designed with increasingly smaller feature sizes pack more functions onto a single component and pave the way toward system level integration. The trend toward "system-on-a-chip" calls for multiple chips comprising a system to be manufactured as a single component. The amount of data collected will mushroom when these integrated systems on a single chip must be measured and tested to ensure product quality and reliability. Next generation technology based on platforms for 300mm wafer sizes and smaller circuit dimensions will require manufacturers to review the viability of their current system infrastructure for analyzing data.

Due to these advancements in technology, the amount of data collected will increase exponentially. It is critical that a strategy for data analysis is adopted which provides appropriate data organization and storage to facilitate data analysis and problem solving. This includes a common framework for easily accessing, managing, and analyzing data. Data, which is the basis for continuous process improvement, reliability, and yield management, is on the brink of explosion.

Today, the major problems in analyzing the data can be attributed to the volume and dynamic state of the data. The dynamic state of the data is due to the rapid pace at which new technology is introduced into the manufacturing process. As this occurs, a significant number of new measurements and test results must be collected and accommodated in the database.

To solve the problem, semiconductor companies must employ the latest techniques for data warehousing and adopt a strategic roadmap for data analysis and decision support. The root cause of the data analysis problem is not unique to the semiconductor industry. Most industries have experienced the same problem and realized the problem lies in the integration and organization of the data. A successful foundation for data analysis and decision support is based on data warehousing concepts and methodologies.

Defining Data Warehousing

Data warehouses are defined as databases designed exclusively for decision support and data analysis. The data warehouse is built to support information access, and typically integrates data from multiple sources. The data is cleaned and restructured to optimize data analysis and decision support. Data Marts are a subset of a data warehouse that focuses on one or more specific subject areas. Data marts are distinguished from data warehouses by size or data content. Data marts are often though of as data warehouses less than 100 gigabytes in size; data warehouses can approach the tens and hundreds of terabytes. Hereafter, references to data warehouses include data marts.

The thought processes and techniques used to define efficient data warehouses are very different from those used in traditional relational database design. The primary use of a data warehouse database is to support the decision making process through data analysis. The design of a data warehouse is intentionally optimized to provide the best retrieval access possible. These databases are read-only and non-volatile, with updates scheduled as batch processes on a periodic basis.

The load process for these databases take on the additional responsibility of transforming, scrubbing, and summarizing the data before it is loaded into the data warehouse. By taking advantage of these techniques, a properly implemented data warehouse can overcome problems associated with such a large amount of data.

Due to the nature of the semiconductor industry, new data elements are continuously introduced to the data collection process and must be added to the database. This presents interesting problems for the underlying design of the data warehouse. A semiconductor fab manufacturing leading edge devices may collect the results of more than 20,000 different measurements or tests. Each time new technology is introduced, up to several thousand additional new data elements must be added to the database to store the results of the new tests and measurements. From an analytical perspective, one would expect the values of these new data elements to be stored in newly created columns that can be easily selected for analysis from the appropriate table in the data warehouse. However, the ability to easily modify and maintain the database at the pace new data elements must be added is a daunting task. In addition, relational database management systems such as Oracle are limited to a maximum of 256 columns in a single table. The number of tables required to accommodate the different tests and measurements can become quite significant.

There are four typical solutions to this ‘dynamic schema’ problem. They are the following:

- Vertical schema
- Multiple table schema
- Rebuilt schema
- Joined schema

Each of these solutions, along with their strengths and weaknesses are examined in the following paragraphs.

Vertical Data Model

The vertical data model is extremely popular in the semiconductor industry and is employed by virtually every Manufacturing Execution System (MES) or Work in Process (WIP) system in use today. Commercial solutions such as Promis® and Workstream®, as well as most homegrown systems use this data model. A WIP system is an operational system designed around the requirements for on-line transaction processing. In addition to these systems, many semiconductor companies choose to store parametric measurements based on the vertical data model. This model is defined as one in which multiple rows of a table are used to store data for a single event or granular transaction. A given column is used to determine which event took place or measurement was taken and a second and possibly additional columns are used to store information about the event or value of the measurement taken.

The table in Figure 1 is an example of a vertical schema based on parametric data. In this example, the table contains the following columns: lot, wafer, die, date/time, parameter name, and parameter value. Each time measurements are taken on a specific die; one row is created in the database for each measurement taken. If 500 measurements are taken of each die sampled, then each sample is represented by 500 rows in the database.

The table in Figure 2 is a WIP example using the same vertical schema. This table would contain the following columns: lot, wafer, process step, beginning date/time, ending date/time, and equipment id. Each time a wafer is processed through a particular step, a new observation is created in the WIP system. From a WIP perspective, this is extremely logical and a good design.

The vertical schema model as used in these two cases is extremely flexible and allows new parametric measurements and new process steps to be created without affecting the database schema. From a loading and maintenance point of view, this is a simple and straightforward model. Unfortunately, the primary purpose of the data is for data analysis and decision support. From this point of view, problems began to surface when there is a need to analyze more than one parameter at a time, or information from more than one process step at a time. In order to analyze information about a given parameter, a subset must be specified which names the parameter to be extracted. In order to analyze two parameters, two separate extractions must be made; the data must be transposed and then joined together. If an extensive thirty-parameter analysis of variance (ANOVA) or multivariate analysis is desired, 30 extractions, transpositions, and joins must take place, requiring multiple levels of joining. The same holds true for analyzing WIP data from multiple process steps.

This model provides easy loading and extremely dynamic data accommodation, which is crucial in situations when a very large number of different measurements are taken and many new measurements are added on a regular basis.

Most analysis will require data transposition. If the primary use of the data is for analysis, then a significant amount of processing time will be spent transposing data repeatedly. Alternatively, if new data columns are added at a significant rate, the overhead and processing involved in administering and rebuilding the schema will prove to be overwhelming. This model can be quite effective if the vertical tables are structured properly and the proper tools are available to ease the data extraction process. Consideration must be given to the tradeoffs between repeated transposition for analysis and the administration of continuously rebuilding the schema.

Multiple Table Data Model

Another data model used to solve the dynamic schema problem is the multiple table data model. In this model, a wide schema is built with all the columns needed for analysis e.g. lot, wafer, die, parameter 1, parameter 2, parameter 3, etc (see Figure 3). When new types of data need to be collected, a new table is created with all the old columns, as well as the new columns (see Figure 4). Data ceases to be populated in the old table and is now populated into the new table. This is an extremely efficient transition for dynamic data, however this model contains two major drawbacks.

The first problem is keeping track of which table covers a particular time period and choosing the proper table for the desired analysis. The second problem occurs when trend analysis is desired over a time period spanning more than one table. If the desired analysis can be extracted from a single table, this model provides a very efficient extraction. When the analysis requires multiple tables, concatenation must occur. Most database management systems are not optimized for concatenation, making it a time-consuming operation. Concatenation, from an analysis perspective, combined with the user frustration of trying to keep up with multiple tables makes this data model less than desirable.

Rebuilt Schema Data Model

Like the multiple table data model, the rebuilt schema data model begins with a wide schema containing all the columns necessary for data analysis (Figure 5). In this data model, when new types of data are collected, the new columns are added to the existing schema (Figure 6). This provides the most efficient table for data extraction and analysis.

The drawback to this data model is the overhead cost associated with maintenance and administration of the

table. It may also become difficult or even impossible to administer the table due to the volume and frequency at which new columns must be created. A second drawback to this data model is sometimes referred to as the sparse matrix problem. The sparse matrix problem occurs when some columns are not populated for a given record thus creating empty spots or holes in the data table. If the entire table were viewed as a matrix, it would appear sparsely populated. If the sparse matrix problem exists, disk storage can be wasted. The best way to solve the sparse matrix problem is to rely on the DBMS's internal compression algorithms. The costs incurred in rebuilding the table as the schema changes can be far overshadowed by the costs of transposition or concatenation described in the two previous data models. This data model is most effective when the schema will remain reasonably static. The rebuilt schema data model should be the first choice in these situations.

Joined Table Data Model

This data model is essentially a hybrid of the rebuilt schema data model. Like the previous two data models this model begins with a wide schema containing all columns desired for analysis (Figure 7). When the cost to rebuild the table becomes prohibitive and schema additions are necessary, a new table is created containing only the key columns from the first table and the new columns (Figure 8). As data is collected, a new record is added to the original table with the columns contained therein and a corresponding record is added to the new table with the same key columns and the new columns. As further schema changes are needed, the new table's schema is modified, and the table is rebuilt until its size makes the rebuild process prohibitive, at which time a third table is created and the cycle continues. Because each table contains the key columns, these tables can be easily and efficiently joined. If analysis is desired which requires columns from multiple tables, a simple inner join is performed in order to extract the data. Most database management systems have highly optimized inner join algorithms. This data model includes all the advantages of the rebuilt schema data model and allows for growth beyond the point where rebuilding a table is cost effective. The drawback of this schema occurs when the schema becomes so dynamic that the administration overhead is prohibitive or impossible.

Attributes of the Data Warehouse

As stated earlier, the attributes that define a good data warehouse are not the same as those that make a good OLTP database. Similarly, the attributes required of a good data warehouse DBMS are not necessarily the same as those of a good OLTP DBMS.

As database management systems have evolved, from their infancy on mainframe computers in the late 1960s, through the relational DBMS of the 1980s, to the latest data warehouse DBMS, some attributes of the database management systems have remained consistent while others have changed dramatically.

The primary attributes to look for in a data warehouse DBMS are optimized retrieval through specialized indexes, optimized loading algorithms, compression/decompression algorithms optimized for speed, and the ability to handle extremely large and wide data tables. Traditional features of a DBMS, such as rollback/rollforward recovery capabilities, extensive garbage collection, reuse of space, and optimized random write capability, are not important for data warehouse applications. These attributes, which are associated with OLTP systems, typically do not pose a problem unless they impede performance or create an unnecessary administrative burden. The data warehouse return on investment is very high in every industry segment. This is especially true in the semiconductor industry where a small increase in yield can have an enormous impact on the bottom line.

With today's connectivity solutions, a site could easily have a DBMS for data warehouse needs and another for their production database needs. Due to the fact that it is often desirable to join data from different data warehouses, it is usually best to choose a single DBMS that is most appropriate for a site's data warehouse needs.

Maestro software

Given the previous considerations, a user must still have a clear understanding of how each of the tables of a data warehouse should be integrated. One must also have the skill and expertise to write the necessary SQL code for extracting the data, before any data analysis can occur. In most organizations, it is estimated that over 80% of the data analysis effort are spent gathering and massaging data. To solve this problem and reduce the amount of time spent gathering and massaging data; JJT has developed Maestro, a commercially available set of software applications based upon SAS Institute's SAS® System of software. The SAS System was chosen for the implementation of Maestro because of its leadership position as the premier statistical analysis tool. The system also has excellent data manipulation capabilities, and an extremely robust object oriented application development (OOAD) environment.

Maestro consists of three major components, the Analysis Reporting Tool, the Output Maestro, and the Maestro Metabase and Metadata Server. The Analysis Reporting Tool (ART) is a drag-and-drop interface to the SAS System's statistical capabilities as well as many specialized objects for yield analysis, trend charts, and simple and composite wafer maps. The Output Maestro is a drag-and-drop interface to the SAS System's extensive output capabilities. These capabilities include printing, e-mail, FTP, creating and exporting delimited files, web enablement, and

seamless integration with specific System tools, e.g. SAS/INSIGHT®, and external tools, e.g. Microsoft® Excel.

The Maestro Metabase and the Metadata Server provide users with drag and drop interaction and automatic code generation during the extraction process. The metabase is a repository of information about all the data sources and columns available for data analysis. The metabase also provides capabilities for defining functions, specifying yield criteria, and split groupings as virtual data columns. Based upon the data columns selected for analysis, the metadata server will automatically generate the optimized SQL code necessary to do any joins and extractions needed for a given analysis. Additionally, the metadata server presents data from the vertical schema model to the user as a wide schema and automatically transposes the data upon extraction. Maestro Version 4 employs a metadata server capable of automatically concatenating multiple tables of the multiple table data model. The Maestro Metabase and Metadata Server virtually eliminate the 80+% time typically spent gathering data. Combined, each of these components performs as an integrated system and represents a complete data warehouse exploitation environment.

Conclusion

The time and effort placed into data analysis and decision support endeavors will become increasingly crucial to the semiconductor manufacturer over the next few years. If the proper tools are chosen and the proper data design is put into place, money spent on these endeavors will be quickly recovered by reducing costs, increasing yield, and accelerating the learning process with new technology. It is critical that a data warehouse approach is taken and the appropriate data models are chosen based on the manufacturer's data analysis requirements. Traditional database management systems and data models will prove to be prohibitively expensive and frustrating without the proper tools in place. However, with the proper solution, data analysis will take only minutes, which in the past took weeks.

Authors

The authors are available to answer any questions at the addresses below.

John T. Stokes
JJT Inc.
1610 West Avenue
Austin, TX 78701
Phone: 512.413.9994
Email: johnt@jjt.com
www.jjt.com

Roger Thompson
JJT Inc.
912 Clermont Avenue
Dallas, TX 75223
Phone: 214.320.2200
Email: roger@jjt.com
www.jjt.com

Trademark Information

APPENDIX

Figure 1 - Vertical Data Model - Parametric Table

LOT	WAFER	DIE	DATE_TIME	PARM_NAME	PARM_VALUE
75024	24	15	12-DEC-97:12:23	PARM_1	2401.003
75024	24	15	12-DEC-97:12:23	PARM_N	1520.005
75024	24	38	12-DEC-97:12:25	PARM_1	2497.563
75024	24	38	12-DEC-97:12:25	PARM_N	5316.593

Figure 2 - Vertical Data Model - WIP data

LOT	WAFER	STEP	EQUIP_ID	PARM_ID	PARM_VALUE	BEGIN_TIME	END_TIME
75024	24	3250	ETCH_05	2304	3421.032	12-DEC-97:23:35	12-DEC-97:24:53
75024	25	3250	ETCH_05	2042	3923.783	12-DEC-97:24:23	12-DEC:97:25:11
75025	01	4600	STEPPER_02	1932	1936.381	12-DEC:97:23:35	12-DEC-97:24:45
75025	02	4600	STEPPER_02	8217	3492.385	12-DEC:97:23:03	12-DEC-97:24:35

Figure 3 - Multiple Table Data Model - Table_1 of Multiple Tables

LOT	WAFER	DIE	DATE_TIME	PARM_1	PARM_2	PARM_N
75024	24	15	12-DEC-97:12:23	2401.003	251.152	672.113
75024	24	15	12-DEC-97:12:23	1520.600	485.445	431.443
75025	24	38	12-DEC-97:12:23	2497.015	731.348	657.824
75025	24	38	12-DEC-97:12:23	5316.385	371.940	795.454

Figure 4 - Multiple Table Data Model -Table_2 of Multiple Tables

LOT	WAFER	DIE	DATE_TIME	PARM_1	PARM_2	PARM_N	PARM_N+1	PARM_N+2
75065	1	15	18-JAN-98:18:37	6482.782	468.652	997.664	795.446	988.651
75065	1	15	18-JAN-98:18:37	7254.457	731.926	461.154	467.315	454.234
75066	1	38	18-JAN-98:18:37	4627.723	467.621	647.934	755.664	997.656
75066	1	38	18-JAN-98:18:37	4831.912	312.664	754.631	467.441	504.774

Figure 5 - Rebuilt Schema Data Model - Initial Table

LOT	WAFER	DIE	DATE_TIME	PARM_1	PARM_2	PARM_N
75024	24	15	12-DEC-97:23:35	2401.003	251.152	672.113
75024	24	15	12-DEC-97:23:03	1520.600	485.445	431.443
75025	24	38	12-DEC-97:23:35	2497.015	731.348	657.824
75025	24	38	12-DEC-97:23:03	5316.385	371.940	795.454

Figure 6 - Rebuilt Schema Data Model - Table_1 Modified with New Parameter Columns

LOT	WAFER	DIE	DATE_TIME	PARM_1	PARM_2	PARM_N	PARM_N+1	PARM_N+2
75024	24	15	12-DEC-97:23:35	2401.003	251.152	672.113		
75024	24	15	12-DEC-97:23:03	1520.600	485.445	431.443		
75025	24	38	12-DEC-97:23:35	2497.015	731.348	657.824		
75025	24	38	12-DEC-97:23:03	5316.385	371.940	795.454		
75065	1	15	18-JAN-98:37:35	6482.782	468.652	997.664	795.446	988.651
75065	1	15	18-JAN-98:37:03	7254.457	731.926	461.154	467.315	454.234
75066	1	38	18-JAN-98:37:35	4627.723	467.621	647.934	755.664	997.656
75066	1	38	18-JAN-98:37:03	4831.912	312.664	754.631	467.441	504.774

Figure 7 -Joined Table Data Model - Table_1

LOT	WAFER	DIE	DATE_TIME	PARM_1	PARM_2	PARM_N
75024	24	15	12-DEC-97:23:35	2401.003	251.152	672.113
75024	24	15	12-DEC-97:23:03	1520.600	485.445	431.443
75025	24	38	12-DEC-97:23:35	2497.015	731.348	657.824
75025	24	38	12-DEC-97:23:03	5316.385	371.940	795.454
75065	1	15	18-JAN-98:37:35	6654.781	452.654	754.654
75065	1	15	18-JAN-98:37:03	6721.844	664.754	766.745
75066	1	38	18-JAN-98:37:35	7625.323	476.664	872.417
75066	1	38	18-JAN-98:37:03	7761.746	346.874	856.744

Figure 8 - Joined Table Data Model - Table_2 Containing Key Columns and New Parameter Columns

LOT	WAFER	DIE	DATE_TIME	PARM_N+1	PARM_N+2
75065	1	15	18-JAN-98:37:35	795.446	988.651
75065	1	15	18-JAN-98:37:03	467.315	454.234
75066	1	38	18-JAN-98:37:35	755.664	997.656
75066	1	38	18-JAN-98:37:03	467.441	504.774