The Report Committee for Scott Allen Collier
Certifies that this is the approved version of the following report:

Semiconductor Manufacturing Dashboard

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Kathleen Barber

Thomas Graser

Semiconductor Manufacturing Dashboard

by

Scott Allen Collier, B.S.Comp.Sci.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

December 2012

Abstract

Semiconductor Manufacturing Dashboard

Scott Allen Collier, MSE

The University of Texas at Austin, 2012

Supervisor: Kathleen Barber

The semiconductor manufacturing process is a complex process that can consist of hundreds if not thousands of steps. During this process an enormous amount of data is generated and collected by several different systems. Analyzing this data can be complicated and time consuming. But, in order to optimize the manufacturing process, it is important to be able to process data quickly and provide data consumers an easy, meaningful way to view the data. Data consumers at a management level need to view data differently than someone who works in the semiconductor fabrication plant (FAB) operating the manufacturing equipment or a maintenance technician who fixes and maintains the equipment. So, it is important to provide these different data views to the users in a logical, organized way. This paper will discuss what a dashboard is, an overview of the semiconductor manufacturing process, and one implementation of a dashboard for the semiconductor industry, the Semiconductor Manufacturing Dashboard (SMD). An explanation of the systems involved in collecting and loading the data, the database structures, and the web servers used for development and production will also be discussed.

Table of Contents

List of Tables

List of Figures

INTRODUCTION

The semiconductor manufacturing process is a complex process that can consist of hundreds of steps which is conducted in a semiconductor fabrication plant (FAB). During the manufacturing process an enormous amount of data is generated, collected and stored. Analyzing this data can be overwhelming, complicated and time consuming. In order to optimize the manufacturing process and to insure that equipment and processing steps are operating within acceptable bounds, it is important to be able to process this data in a meaningful way.

Throughout the day reports are generated in preparation for upcoming meetings. In addition, managers and engineers run either standardized reports created by a reporting group or reports that they have created themselves by using software applications so that they can monitor production numbers and the overall state of manufacturing systems. Engineers and management also enter data collected from these reports into Excel spreadsheets to help make sense of the data.

Although a FAB may have standardized reports, each data consumer may have different needs and, therefore, need to view different data. FAB managers will need to see an overall view of the FAB and care about when products can be shipped, if they are meeting shipping deadlines, and what is the mix of devices being manufactured. Department managers need to see how many items have been processed in their department, how many items are coming to their department, the status of the manufacturing equipment that is in their department, or how other departments are doing. Equipment operators may only care about work coming to their machine how much work they have done during their shift, or if any preventative maintenance is upcoming. And, equipment maintenance technicians may only want to see information about the tools that they are responsible for taking care of. It is important to provide these different data views to the users in a logical, organized way.

This paper will discuss what a dashboard is, an overview of the semiconductor manufacturing process, and one implementation of a dashboard for the semiconductor industry, the Semiconductor Manufacturing Dashboard (SMD). An explanation of the systems involved in collecting and loading the data, the database structures, and the web servers used for development and production will also be discussed.

DASHBOARDS

Traditionally, data has been presented in the form of tables whether in print or in Excel spreadsheets. While these are still useful and commonly used, sometimes data in these formats makes it difficult for users to obtain the overall view of what the data represents or to spot trends in the data. Also, many times, the data represented on printed pages and Excel spreadsheets is static and already out of date by the time the data is being examined. This is where a dashboard can be useful.

Dashboards provide data in a graphical format that can help users obtain a complete overview with just a glance. A well designed dashboard can take a large amount of data and present this data in a way that is meaningful to users. Dashboards are generally interactive allowing users to interact with the data to change how the data is displayed and to drill-down deeper into the data. The data that is used to create the dashboard can be static or updated in real-time depending on the purpose of the dashboard. Figures 1 – 3 show examples of some common dashboards.

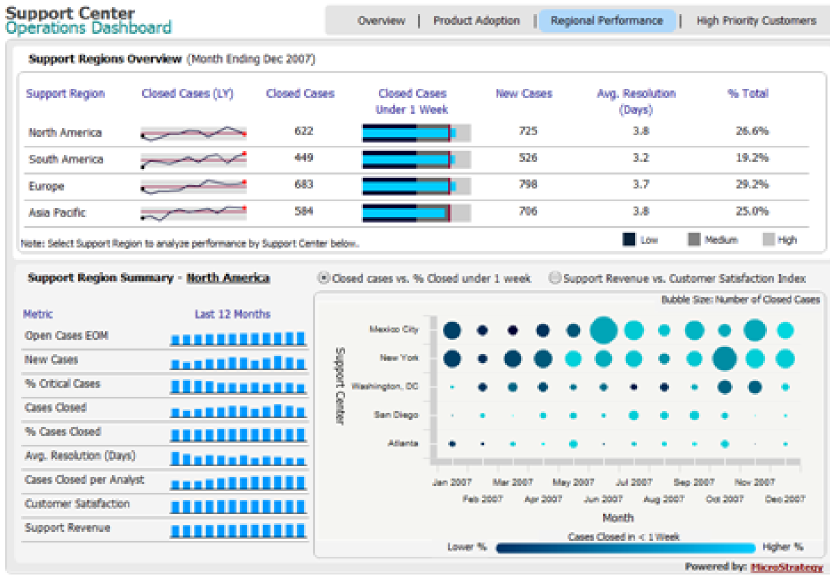

Figure 1: Dashboard by Microstrategies [1]

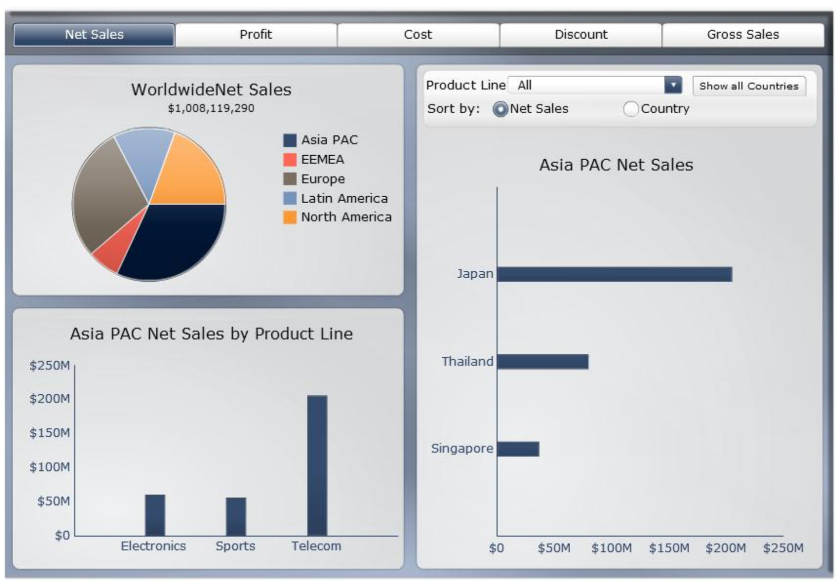

Figure 2: Sales dashboard from Centigon Solutions [2]

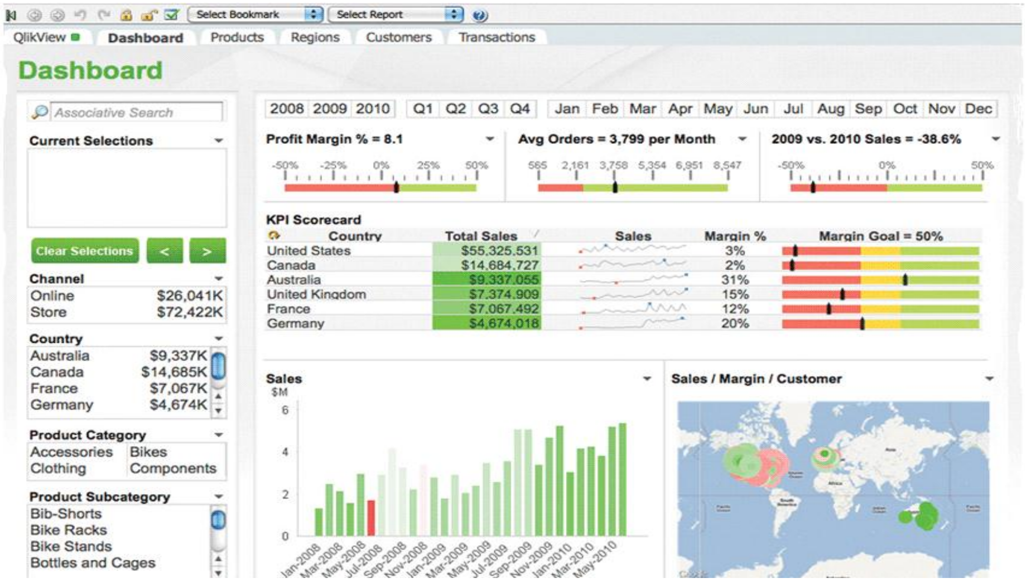

Figure 3: Dashboard by Qlikview [3]

After examining many dashboards, most dashboards focus on sales, operations and logistics, financial analysis, inventory levels, and market analysis. Dashboards that focus on manufacturing generally focus on the areas above and not on the activities happening on the manufacturing floor and are not designed to monitor the actual manufacturing process. All dashboards rely on the data being formatted and organized so that the dashboard can display the information properly. Therefore, it can be difficult to force existing dashboard products to display the data as needed. Accomplishing this requires developers with a strong understanding of the data to create dashboards either by writing custom code or by using tools provided by the companies selling the dashboards. Experience with Qlikview has shown that this can be time consuming, difficult, and requires special training and a complete understanding of the data. With regards to the semiconductor industry, it is difficult to find individuals that have enough semiconductor experience to understand the data and enough programming skills to create a dashboard.

The decision to purchase and modify a preexisting dashboard solution and to learn an API be learned to display data useful to clients in the semiconductor industry was discussed in company meetings. The following items were examined to determine if the

SMD should be developed as a product: that company employees have worked in over 10 major semiconductor companies and foundries around the world and have not seen a solution that was comparable to the SMD product but had seen several attempts to use premade solutions fail; that numerous requests for a semiconductor dashboard solution were made from several clients; and, that both developing a product and modifying an existing product takes time and money. The decision to develop the SMD was made easier when a major semiconductor foundry agreed to pay for some of the development costs of the SMD in order to have the SMD in their facilities. Before examining the SMD, it is necessary to provide an overview of the manufacturing process and explain some common terms used in the semiconductor industry.

SEMICONDUCTOR MANUFACTURING

Semiconductor manufacturing is a complex process that consists of hundreds of manufacturing steps called "wafer fabrication." Throughout this process, bare silicon wafers are repeatedly processed on tools by adding layers of material that are patterned into integrated circuits. During the wafer fabrication process, approximately 20 to 30 layers are placed on the bare silicon wafers. Once all the layers are complete, the wafer consists of many completed integrated circuits. Completed integrated circuits can contain millions of simple circuits made up of transistors, diodes, capacitors, resistors, etc. that can work together to perform the complex functions needed to create Microprocessors, Memory, Digital Processing, etc. The type of item that is being created is called a "device." Each complete integrated circuit is called a "die." Once the integrated circuits are completed, they are cut into individual chips that are then packaged for use in various electronic devices. In order to provide control and monitoring of the manufacturing process, data is collected at each tool and at most of the manufacturing steps. This data is analyzed by engineers so that the final product can meet certain specifications.

Silicon Wafer Preparation

A silicon wafer starts out as sand. Sand contains a large percentage of silicon that serves as an excellent semiconductor. The sand is first collected, purified, melted, and then cooled into a solid cylinder. The resulting cylinder is called an "ingot." The process of producing an ingot is done in an extremely clean environment so that the ingots produced and free of electrical and physical contamination. The diameter of the ingot determines the size of bare silicon wafers that can be produced. Currently, a common diameter for wafers is 300mm (approximately 12 inches). The ingot is sliced in to thin wafers that are approximately 1mm thick.   After being sliced, the edge of each layer is rounded and polished into a mirror-smooth surface. Once this has been completed, a thin layer of oxide is grown on the wafer before entering the semiconductor manufacturing process. See Figure 4 for an overview of the process for preparing a semiconductor wafer.

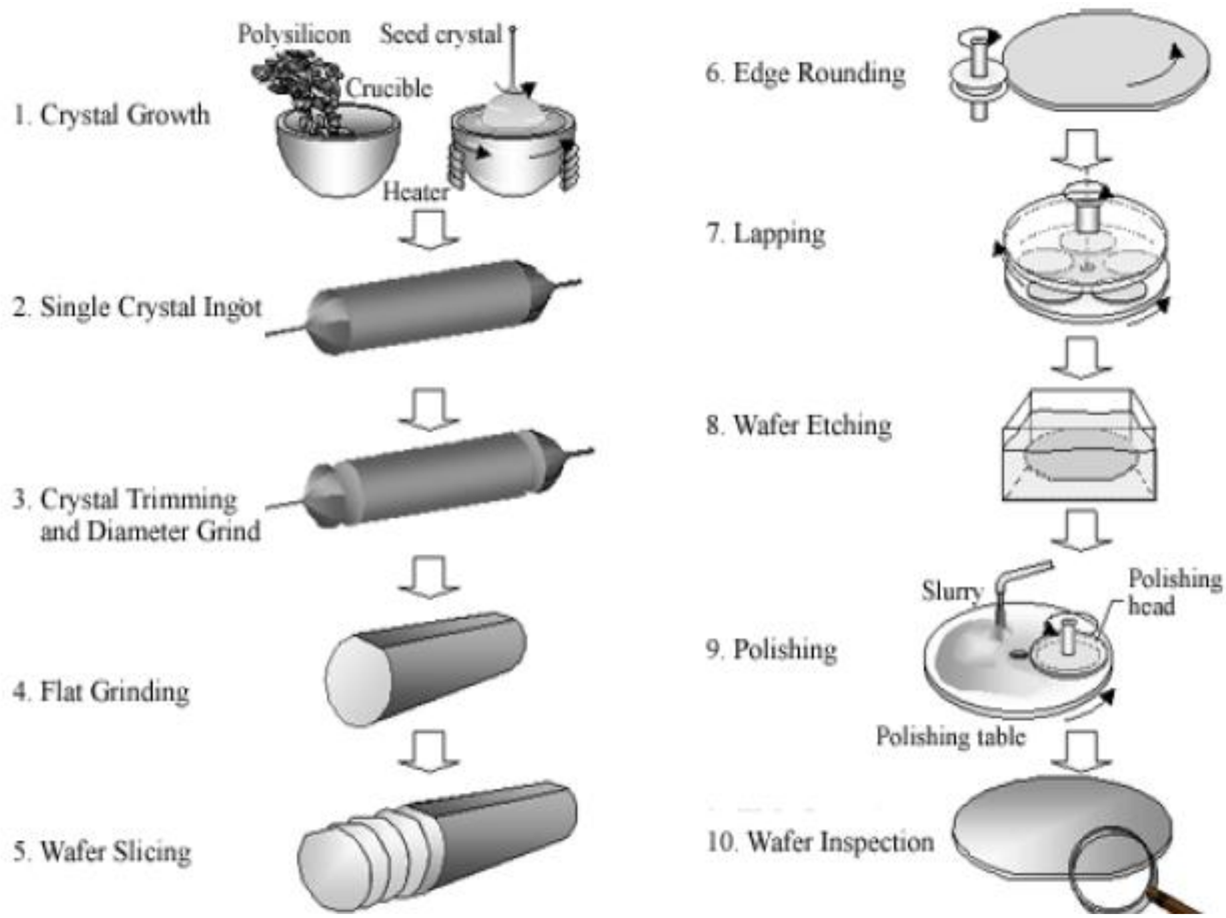

Figure 4: Creating Silicon Wafers Process [4]

Semiconductor Modules

As stated earlier, the series of steps taken to manufacture semiconductors is called “wafer fabrication.” The place where semiconductors are manufactured is referred to as a “fab.” The manufacturing process consists of hundreds of complex steps that can be broken down into different process groups. These process groups are referred to as "modules." Although some semiconductor manufactures transport wafers one at a time through each process, it is more common for wafers of the same type to be placed in group of approximately 25. This group of wafers is called a "lot." A lot will go through a module several times in depending on the type of semiconductors being manufactured and measurements taken at each step. The order in which the steps are taken also depends on the desired output and is called the lot's "route." An overview of the general process flow and the relationship between modules can be seen in Figure 5.

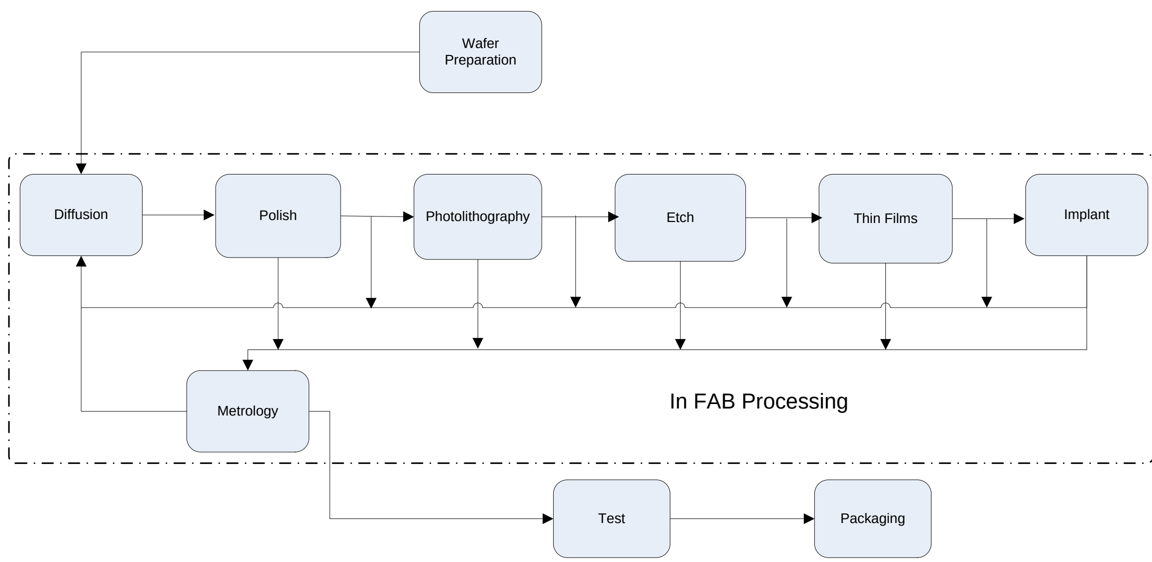

Figure 5: General Semiconductor Manufacturing Flow

DIFFUSION

The diffusion process uses furnaces to grow oxidation layers on the wafer. According to Quirk and Serda, these layers are used to [4]:

1. Provide device scratch protection
2. Isolate any contamination that can occur

3. Neutralize the electrical properties of the surface
4. Provide a dielectric material for gates being built on the wafer
5. Provide a barrier from other chemicals that have been added to the wafer
6. Provide a dielectric layer between metal conductor layers

POLISH

The polish processes planarizes the topmost layer to remove high spots and prepare the wafer for the photolithography layer. This can be done mechanically or chemically. As area of a wafer is built up and etched away for each layer, a layer can become uneven. This can cause sub-micron mountain ranges on the top most layers. Because photolithography equipment uses lenses to place patterns on the wafers, these mountain ranges can cause focus errors which can cause inconsistency in the patterns. If measurements fall out of range, the device being produce may either not work at all or have degradation in performance. It is important that each new level start on a level surface to apply photoresist on it.

PHOTOLITHOGRAPHY

Photolithography, commonly called “photo” or “litho”, uses masks or reticles to place patterns on the wafers. Although sometimes the terms mask and reticle are used interchangeably, a mask contains patterns that make up a larger device that can contain 100’s of integrated circuits. A mask contains the pattern for all or part of a layer that covers the entire silicon wafer. On the other hand, a reticle is smaller and only contains layer patterns for a few integrated circuits. Reticles cover only a small part of a wafer. One can think of masks and reticles as lenses that consist of patterns on them that either block the ultra-violet light or allow the light to pass through.

Upon entering the photolithography module, a substance caused photoresist is spun on the silicon wafer. That photoresist is then soft baked on the wafer to keep the photoresist in place. Then, by using a mask or reticle and ultraviolet light, a pattern is placed upon the wafer. The areas of photoresist that are exposed to the ultraviolet light harden protecting the layers below. The areas of the photoresist that are not exposed to the ultraviolet light are soft and are removed. This is develops the wafer. Once the softened photoresist is removed, the dimensions of the photoresist are measured and collected for analysis.

ETCH

Etching is accomplished by using chemicals which is called "wet etching" or by using gases in plasma etchers which is called "dry etching." In either case, the oxide or film material that was left exposed after developing a wafer in photolithography is removed and the remaining photoresist is removed. This leaves a pattern on the wafer for that layer that makes up an integrated circuit. Again, the dimensions of the layer are measured and collected.

THIN FILMS

Thin Films processes deposit a conductive or non-conductive material on the top of the wafer. This material is called a substrate and is used to connect the devices and layers together. The thickness of the layer deposited on the wafer depends on the requirements to build the device specified. After the layer is deposited, both the thickness and the electrical properties of the film layer are measured and collected for analysis.

IMPLANT

The implanting processes can be done either before or after the etching process. Whether implanting is done before or after etching depends on the results that are needed to build a device. The purpose of implant processes is to introduce electrically active ions called dopants to the wafer that modify the semiconductor properties of the materials on the wafer. Implant uses equipment called implanters to add the dopants to specific areas on the wafer. To insure quality, the properties of the layer are measures and collected after the implant process has been completed.

METROLOGY

The metrology module consists of the measurement steps that occur during the fabrication of semiconductors. Although each module takes and collects measurements after each step, steps dedicate to taking and collecting measurements are added to the fabrication process in order to make sure that each device in within specifications.

AUTOMATION

Automation does not have any steps dedicated to it. But, automation is the module responsible for making sure that lots are transported to the correct location as soon as possible. Locations can include stockers, that hold lots while they wait to be processed, manufacturing equipment, and metrology equipment. Instead of relying on manpower to transport lots from one location to another, many FABs use carriers that ride alone tracks to transport lots from one place to another. This system is called the Automated Material Handling System (AMHS).

TEST

Once the wafer has been through all of the steps, the wafer is then set to the Test module. Once wafers enter the Test module, they cannot be returned to the fab for processing. Although wafers are measured and data is collected at each step, the manufacturing process is so complex it is important to test die on the wafer to make sure that the die are operating correctly and within acceptable ranges. Even die that don’t operate in optimum levels may be able to still be sold to consumers. For example, a wafer that has gone through the manufacturing process to build computer processors may produce processors of different speeds. Although all of the die on the wafer are the same device and test correctly for inputs and outputs, they may operate at different speeds. So, one die may be a 3.4 GHz processor while other die on the wafer may be 3.2 GHz or some other speeds.

KEY METRICS FOR SEMICONDUCTOR MANUFACTURING

This section will define only those metrics that are of a primary concern to the SMD and that can help the reader understand the terminology used in this paper. Also, some metrics are defined differently depending on the company. The following definitions are generic in nature.

- **Entity Statuses**: Tools, Chambers, and Port can be referred to as entities. These entities can be in different states and it is important to monitor the states of these entities to make sure that everything is running smoothly. Figure 6 shows the colors used for the SMD.

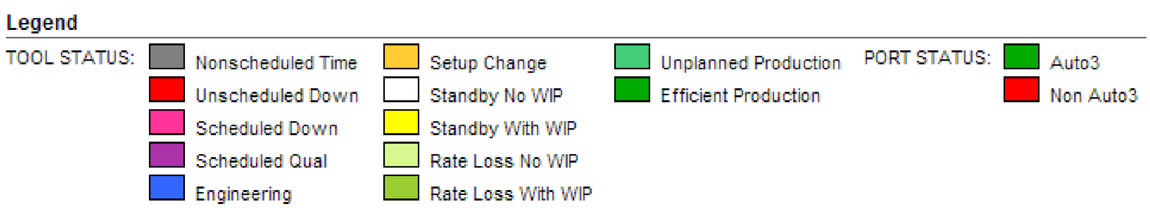

Figure 6: SMD legend for tool and port statuses

- **Holds**: Holds are placed on lots. A hold prevents the lot from being processed until some activity has occurred. Holds may be placed on lots for a variety of reasons.
- **Moves**: Something is counted as a move if it has completed a step and is moved out of that step. Like WIP, moves can be broken down into different ways depending on the FAB.
- **Pace**: Pace is the amount of moves expected to happen during that shift at the current rate of moves that have already occurred since the start of the shift.

- **Preventative Maintenance (PMs)**: Because of the precision needed to manufacture semiconductors and because manufacturing equipment is extremely expensive, it is important for the manufacturing equipment to be maintained. Therefore, it is necessary to perform regular scheduled preventative maintenance.
- **Priority Lots**: Lots can be assigned priorities that can define their level of importance. Lots with higher priorities should usually be processed ahead of lots of lower priorities.
- **Rework**: This is material that has to repeat a step or a series of steps.
- **Scrap**: A scrap is a wafer that can no longer be counted as production due to breakage or some defect.
- **Test Wafers**: Wafers used to test or calibrate manufacturing equipment.
- **Work In Progress (WIP)**: This is material that is currently being manufactured. WIP can be broken down into different types of WIP depending on the needs of the FAB. For example, one FAB may want to monitor WIP broken into production, development, and test wafer WIP while another FAB may want to see WIP broken up into a way more meaningful to that FAB. Figure 7 shows the types of WIP and the colors that represent them on the SMD.

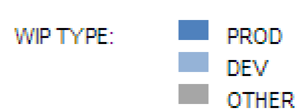

Figure 7: SMD legend for WIP types used

- **WIP Coming**: WIP is counted as coming if it is expected to arrive at a step for a certain module, toolset, or tool before the shift ends. This is an estimated number based on where lots are currently and the average time it takes for those lots to reach the location being examined.

- **Yield**: Is the number of good number of items divided by the total number of items being measured. Yield can refer to lots, wafers, or die and is a percentage.

SYSTEMS

In this next section, an explanation of the systems that are used to collect and store the data produced by the manufacturing process as well as the systems that are used to access, process, and show this data to the data consumers in a meaningful manner in real-time will be discussed.

Machine Execution System (MES)

The MES is a system that is used to manage process conditions, operation and process flow, equipment statuses and performance, and operator actions. An MES can communicate with equipment, accept user input and make decisions based on this input and predefined rules. Data collected from processes, equipment, rules and other necessary data are stored in the MES's database. Some common MES systems are Workstream [5], SiView [6], FAB300 [7], and SMARTFACTORY [7]. Since most MESs are not temporal in nature and only keep track of events as they happen, temporality needs to be added to these events. This is accomplished by storing the events are stored in a database that adds temporality. This database can be used for reporting by either accessing directly or with software applications. Data is entered into this temporal database in real-time and can be replicated across several databases.

Database

The application is currently designed to work with Oracle 10g or 11g. Using Applied Materials' APF Formatter [8] and Activity Manager [9] applications, necessary data is populated into the database used by the dashboard application. The data for the dashboard exists in four different schemas: BASE, INPUT, ADMIN, and APP.

BASE: This schema contains data tables and views that contain data at its most basic level and is used by views in the APP schema. This is data that has come from the

MES and has been formatted and processed using APF Formatter. This data is later further processed by Oracle views to be useful to the dashboard.

INPUT: This schema contains data tables and views that provide input for views in the APP schema. Some of the tables in this schema include:

SHIFTS	Shift start and end times, name, letter, and work week
WEEKS	Work week names, start and end dates, and quarter
EQP_STATES	Tool states, description, and precedence
EQP_TOOLS	Tool name, type of tool, location, and stocker
EQP_TYPES	Tool types, modules that own them, manufacturer, and description
CUSTOMERS	Customer names and the colors that represent them
FACILITIES	Facility key and description.
MODULES	Module key, description, facility, and the colors that represent them
TECHNOLOGIES	Technology names and the colors that represent them
PROCESSES	Process name, descriptions, module that owns the process, equipment types that can run the processes, and recipes for the process
ROUTES	Route name, description, is_production, is_main_route
ROUTES_STEPS	Step name, route the step is part of, and recipe for the route and step

Table 1: INPUT schema tables

ADMIN: This schema consists of tables, procedures, and functions used by scripts run by Applied Material's Activity Manager. The main table, ADM_LOAD, contains a list of Oracle procedure calls, when these calls should be executed, and the order in which they should be called. These procedures calls either completely replace the data in an APP data table or append data to an APP data table by using data results from previously created Oracle Views. These views must adhere to certain naming standards. Also, a table must exists for which ADMIN has read, write, and delete permissions. The views can be in any schema. An example of the two commands can be seen in Table 2.

Frequency	**Order**	**Execution_String**
3 Minute	11	call adm_auto_table_refresh('dash_p_spotlight_opers', 'app')
3 Minute	6	call adm_auto_table_refresh('dash_b_wip', 'app')
30 Minute	10	call adm_auto_table_refresh('dash_p_chart_yield', 'app')
Shift	50	call adm_auto_table_append('dash_p_h_module_summary', 'app')

Table 2: Example of calls used to populate data

Table 2 contains two procedures that will be executed every three minutes, one procedure that will execute every 30 minutes, and one that will execute once at the beginning of every shift. The three minute entry with an order of 6 will call the `adm_table_refresh` procedure that will delete all of the data in the DASH_B_WIP table in the APP schema and call a view call DASH_REF_B_WIP and store the results into the DASH_B_WIP table. The shift entry with an order of 50 will call the `adm_table_append` procedure that will append the results of view DASH_APD_P_H_MODULE_SUMMARY to the DASH_P_H_MODULE_SUMMARY table in the APP schema.

APP: This schema provides all table and views used directly by the application as well as tables that contain data from BASE and INPUT schemas that has been processed for use in the application. All calculations necessary to create the charts and data tables on the web pages are calculated in the views.

The views and tables in this schema have prefixes that allow developers to easily determine the purpose of these tables and views. These conventions are described in Table 3.

DASH_B	Base data that is used by other views and is not accessed directly from web pages.
DASH_B_H	Historical base data that is used by other historical views and is not accessed directly from web pages.
DASH_P	Data that is accessed directly from web pages. These views and tables are only not used by any other views.
DASH_P_H	Historical data that is accessed directly from web pages. These views and tables are only not used by any other views.
DASH_C	Configuration data that is used by component of the web site. For example, DASH_C_CHARTS would contain configuration information about each chart such as chart type, height, and width.
DASH_W	Tables that are read by web pages but can also be written to from web pages. These tables are only not used by any other views.
DASH_REF_B	Views called by ADMIN in order to refresh the base data in a table. These views are only accessed by the `adm_auto_table_refresh` procedure in the ADMIN schema.
DASH_REF_B_H	Views called by ADMIN in order to refresh the historical base data in a table. These views are only accessed by the `adm_auto_table_refresh` procedure in the ADMIN schema.

Table 3: APP schema database table and view prefixes

DASH_REF_P	Views called by ADMIN in order to refresh the data tables that are accessed directly from web pages. These views are only accessed by the `adm_auto_table_refresh` procedure in the ADMIN schema.
DASH_APD_P_H	Views called by ADMIN in order to refresh the historical data tables that are accessed directly from web pages. These views are only accessed by the `adm_auto_table_refresh` procedure in the ADMIN schema.

Table 3 (continued): APP schema database table and view prefixes

Database Loading

Using Applied Materials' APF Reporter, reports are created that obtain the data from the temporal database. These reports contain the logic to process the data from the temporal database and output SQL statements to text files. Jobs are scheduled to run in the Activity Manager every 3 minutes, 10 minutes, and 30 minutes and at the beginning of each shift. At these times, activities happen in the following order (See Figure 8):

1. All reports designated to run at that time are run creating new files containing SQL statements that are used to populate database tables.
2. The script runs the SQL queries listed in the files and populates the data tables in the BASE schema.
3. A UNIX script is executed that queries the Oracle database and returns a listing of Oracle procedure calls that should be executed. For example, every three minutes the script would return a list of Oracle procedure calls that are to be executed every three minutes.

4. The procedures call Oracle views that populate their related database tables.

Error checking is performed when reports are run to create Oracle SQL statements, BASE database tables are being loaded, and APP database tables are loaded. When errors occur, an email is sent to team members to notify them that an error has occurred. If an error occurs while loading data into the APP database tables, in addition to sending an email, each update that has failed is updated to show that that update has failed. An entry into the database is also inserted into the database stating the error that occurred.

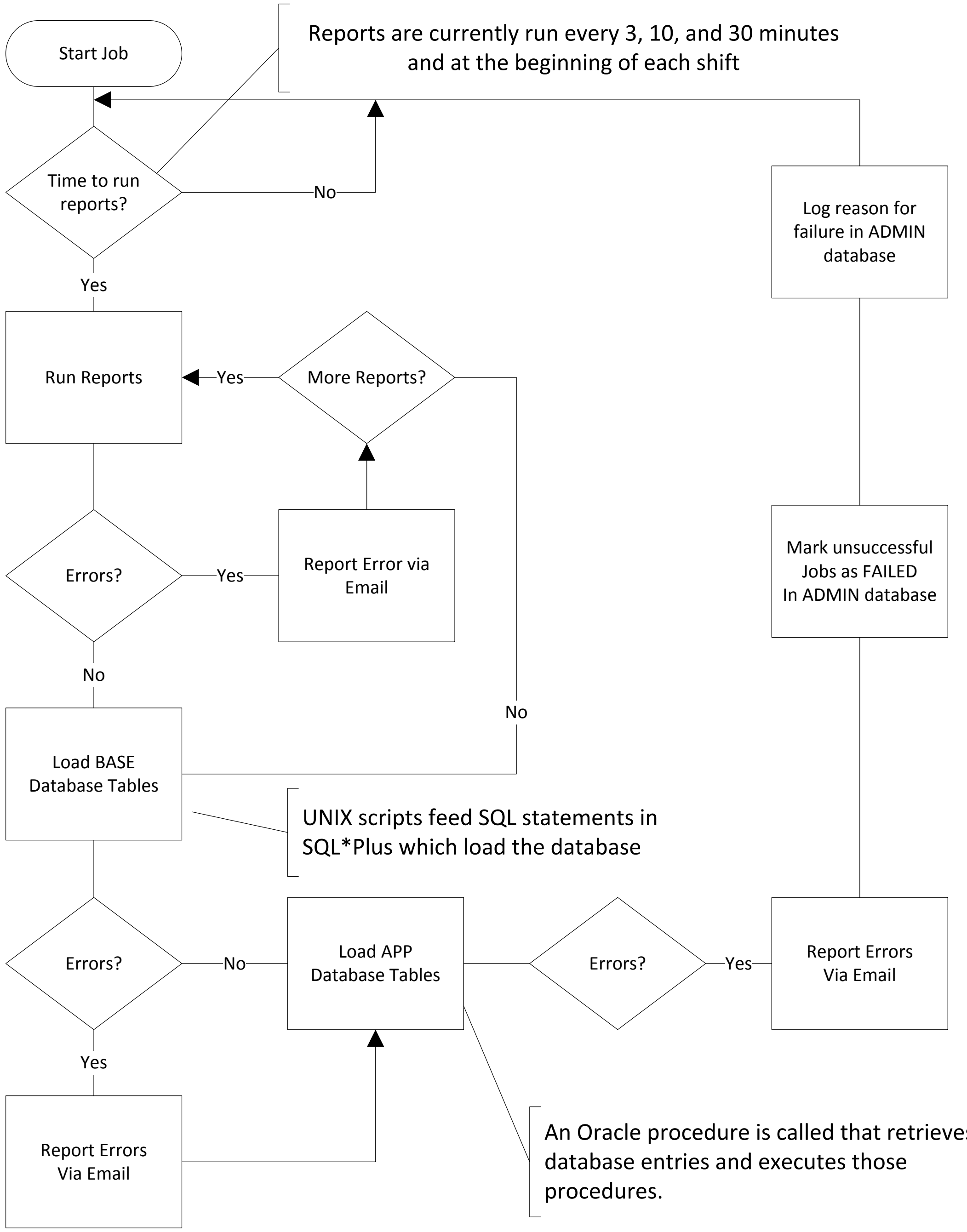

Figure 8: Database loading sequence

Server Hardware

The hardware requirements are those requirements necessary to run the database and the software listed in the software requirements discussed in the next chapter. All of these requirements are the minimum requirements. Higher values are recommended for improved performance and to allow room for growth.

- Processor: 1 GHz (32 or 64-bit)
- RAM: 1 GB
- Hard Drive: 2 GB

An overview of the systems can be seen in Figure 9.

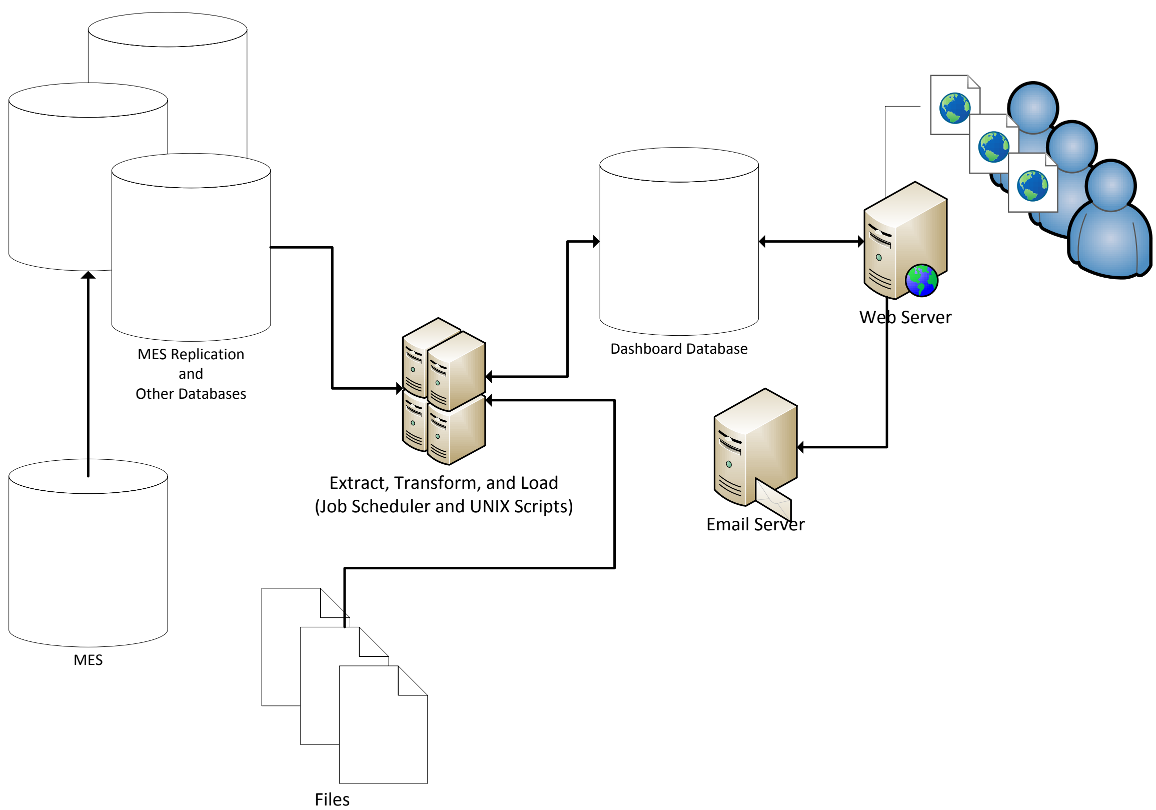

Figure 9: Systems

SOFTWARE

MICROSOFT VISUAL STUDIO

Technically, programming could have been done with a simple text editor. But, to make managing the project easier and to speed development, Microsoft Visual Studio 2010 was used initially. This was later changed to Visual Studio 2012, although this was not required by the applications.

WEB SERVER

Since the SMD application is web-based, a web server is needed to provide enterprise wide access to users. The web server must have the following applications installed and properly configured in order to run the dashboard:

- Windows Server 2008 R2
- Internet Information Server IIS 7.5
- .NET Framework 4.0

LANGUAGES AND WEB TECHNOLOGIES

The SMD was designed using ASP.NET Web Forms programming model using C# as the code that runs on the server, server-side code. For code that runs inside the web browser, client-side code, JQuery and JavaScript is used. All queries to the database are written in Oracle SQL.

Initially, the applications was to be designed using ASP.NET MVC. The ASP.NET MVC programming model is uses the Model, View, Controller model. This model allows developers more control over the output of the HTML and requires the developer to have a strong grasp of HTML, CSS, AJAX, and JavaScript. ASP.NET MVC is easier to setup and conduct Unit Testing which allows for test-driven development [10] which is usually difficult to do on web sites. While ASP.NET MVC offers several advantages, this was a new model and would require time to learn the model before development on the SMD could begin. And, in general, the ASP.NET Web Form model offers faster development initially. So, due to time constraints, ASP.NET Web Forms was used with plans to convert the application to MVC when time permitted.

ASP.NET Web Forms offers a control- and event-based programming model that is familiar to many programmers; has controls that developers can drag onto a page and configure that encapsulate HTML, JavaScript, and AJAX; and, can bind controls to data [10]. This can include panels that developers can configure to update sections of web pages using AJAX called Update Panels. There are also other useful controls such as data grids and charts that can speed up development.

Charts are currently created by either using charting components included with ASP.NET or by creating the charts using CSS and HTML DIV tags.

SEMICONDUCTOR MANUFACTURING DASHBOARD

The SMD application actually consists of three separate IIS applications that work together to provide the functionality needed. These three applications are Common_Resources, Feedback, and SMD applications.

SITE STRUCTURE

All of the applications reside in the web sites root directory. For ease of configuration and management all three applications reside in separate folders and have the same parent directory. Within each of these applications, a directory structure exists to aid in organizing the files into a logical manner. See Figure 10.

COMMON_RESOURCES APPLICATION

The Common_Resources application is an IIS application that contains resources common to the other two applications as well as any other applications that may be developed to work with the SMD or use the Feedback application. This consists of a collection of images, CSS files, and various client-side scripts that can be used by other applications. Initially, all resources were placed within their respective applications. But, it was soon decided that by placing the resources in their own IIS application, a duplication of resources could be reduced. In the Common_Resources application, images, CSS, and script files that are specific to clients i.e. company logos and color schemes are placed in their own subdirectory. Having these resources in its own application also solve some path name issues that occurred with the Feedback application that will be discussed later.

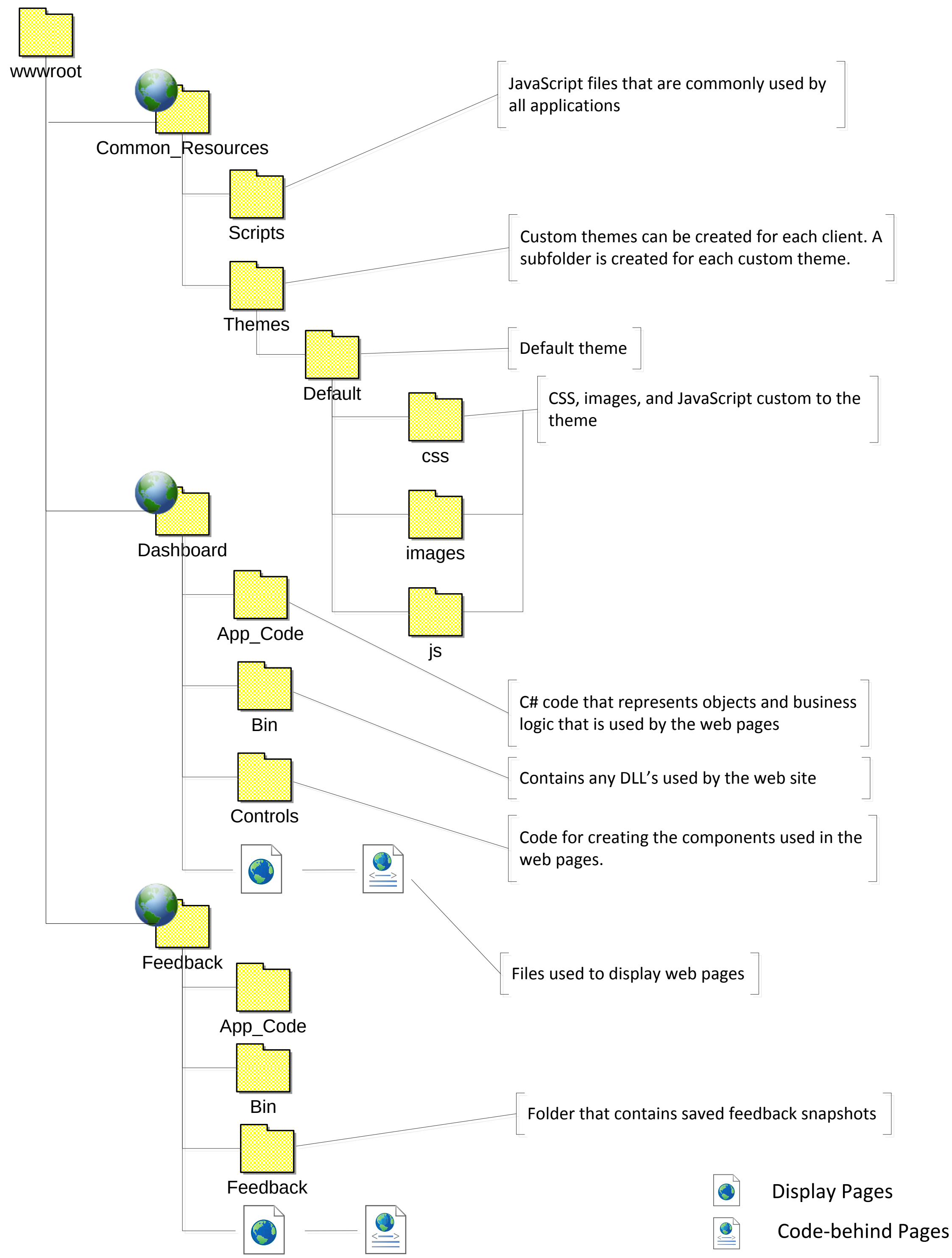

Figure 10: Application and web site structure

FEEDBACK APPLICATION

The Feedback application in an IIS application that consists a collection of client-side scripts written in JavaScript and JQuery that provides user interaction and a data collection functionality as well as server-side C# files that takes the information from the client-side scripts and saves this information to files, logs the information into a log file, and sends email messages to the necessary recipients.

The Feedback application serves two purposes. The first functionality is called "Feedback" and provides feedback and report errors about the application to support personnel and to allow the user to send a copy of the web page in its current state to them. The second functionality of the Feedback application is called "Share a Page" and allows users to send a copy of the web page in its current state to recipients that the user can specify. Users can access the both the "Feedback" and "Share a Page" functionalities from links provided at the top right of every web page shown in Figure 11.

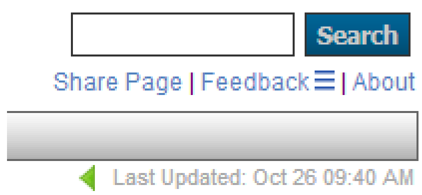

Figure 11: Share A Page and Feedback links

Both the "Feedback" and "Share Page" functionalities operate in the same way. The main differences between the two are the forms that are presented to the web user asking for input and the recipients of the final information. The web user clicks on the link, fills out the appropriate web form, the data from the form and the code that makes up the web page being captured is collected, and the data is sent to recipients. The "Feedback" form can be seen Figure 12.

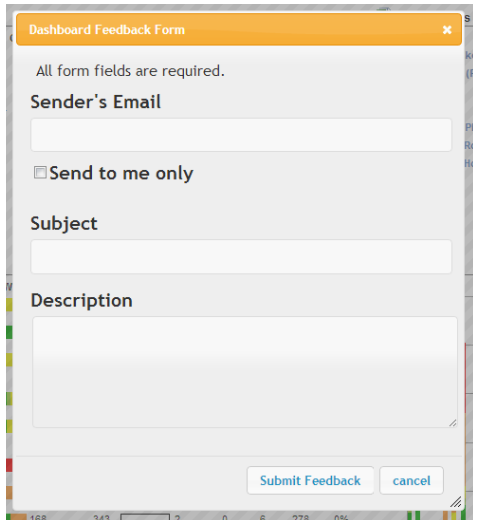

Figure 12: Snapshot of Feedback Form

The “Share Page” form can be seen in Figure 13 and Figure 14 shows the activity flow of the two functions.

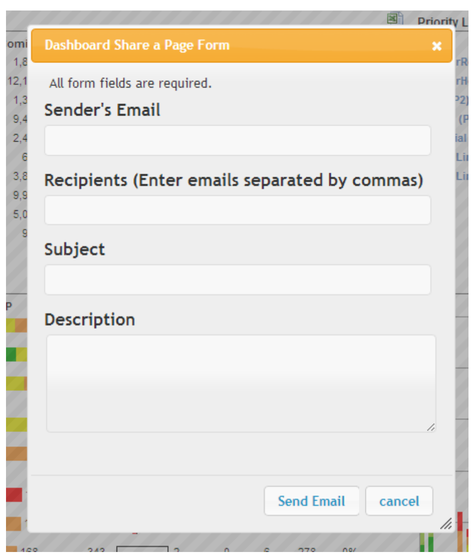

Figure 13: Snapshot of Share Page

When developing the Feedback application, some issues had to be resolved. Although capturing the web page code was relatively easy with the use of JQuery, sending that data in a web form was difficult at first due to IIS security issues that prevented the sending of HTML code in a web form. After some research, it was discovered that this security setting could be turned off in the application's configuration file.

After the security issue was resolved, sending the page code was difficult due to the size of some of the pages' source code. This was resolved by removing section of code that did not need to be captured and saved for later viewing. Some of this source code included web forms in the source code. Although no problems with page sizes has been reported, a method of compressing the page first before sending the page and then uncompressing the page on the server before saving it is currently being researched.

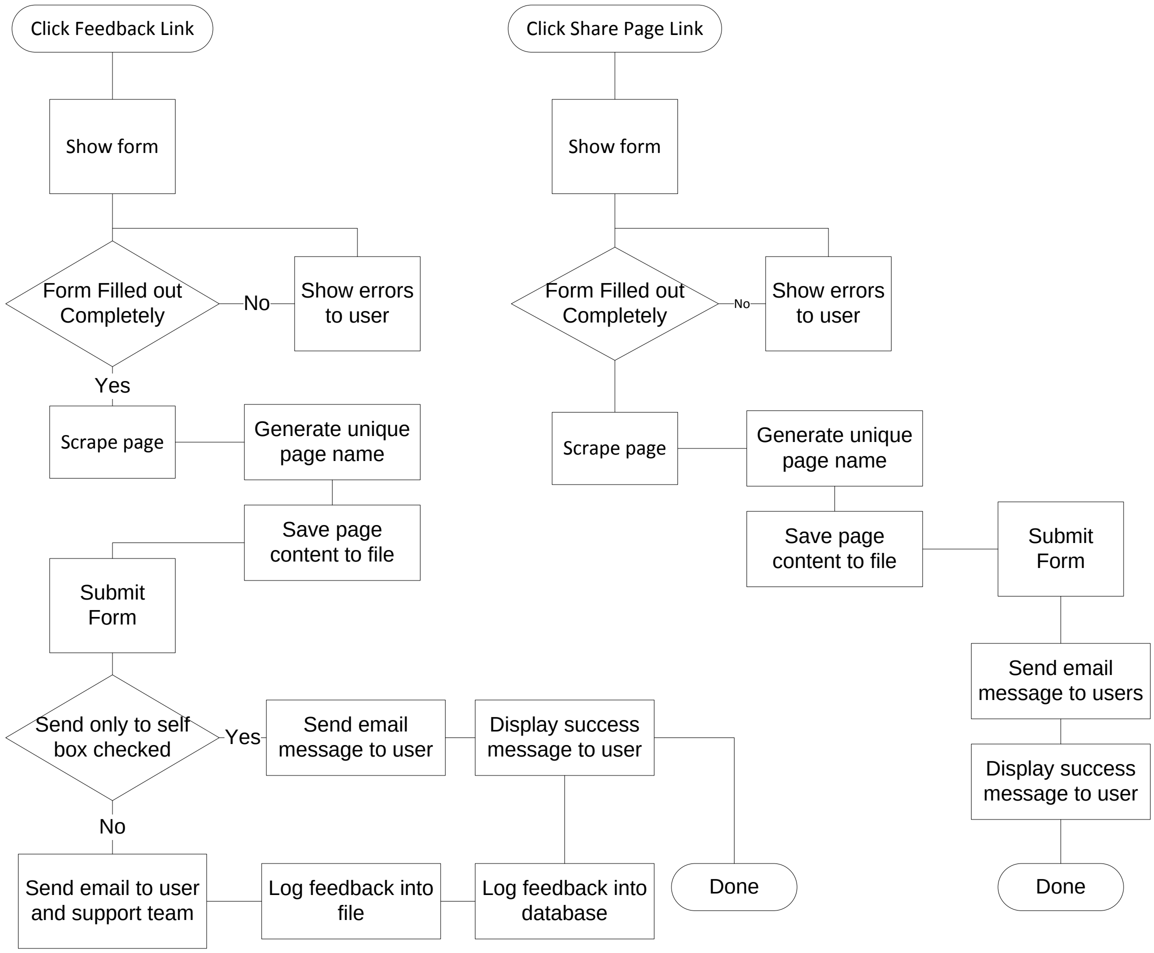

Figure 14: Share Page and Feedback Flow Chart

Once the security and page size issues were resolved, it was discovered that, because the captured pages are stored within the Feedback application, the links to CSS, scripts, and images in the captured pages did not reference the correct location of these resources. This was resolved by creating the Common_Resources application discussed earlier. Placing all the resources in that application solved the link referencing issue because now all applications pointed to resources in a common location.

Once the pages are captured and the links have been sent to the appropriate recipients, the pages need to be viewed by users. Since the link references a snapshot of a web page that contains code that recreates the web pages as they looked at the time the page was captured, it must contains all of the source code need to do that. This includes all hyperlinks. Since the purpose of the page is to only exist as a snapshot of the page, all hyperlinks and buttons needed to be disabled without changing the look of the page. Two main options in solving this problem were: finding all of the hyperlink code and replacing there link with some other HTML tags such as SPAN tags as the pages are captured or disabling the links on the page. The second option proved to be the best because it preserved the pages' source code and, with JQuery, disabling links and buttons was easy. The JQuery code can be seen in Table 4.

```
$(function () {

    /* Returns false when a link is clicked.
       This disables link. */

    $('a').click(function() { return false; });

    /* Disables a line appearing under links */

    $('a').hover(
        function () {
           $(this).css('text-decoration', 'none');
        },
        function () {
          $(this).css('text-decoration', 'none');
       }

    );

});
```

Table 4: JQuery code to disable web links

DASHBOARD APPLICATION

Figure 15: Dashboard snapshots [11]

Layout

It is important that the dashboard have a consistent look and feel throughout the entire application. To accomplish this, a template was created that is the foundation for every page in the SMD application. ASP.NET Web Forms allows developers and designers to easily do this by creating master pages. A master page defines the layout and common elements of a page that can be then by used as a template for other pages. This can include both server and client side code that is common to all of the pages that are based off of the master page. A web site can have as many master pages as needed to facilitate design. After creating a master page, developers and designers can associate a master page with a new Web Form page in Visual Studio by selecting which master page the new page should use. The master page for the dashboard defines the overall layout of the page including defined place holders for the top menu, breadcrumb, and spotlight. It also contains hidden forms needed for the feedback application and all references to CSS and JavaScript files used by the dashboard.

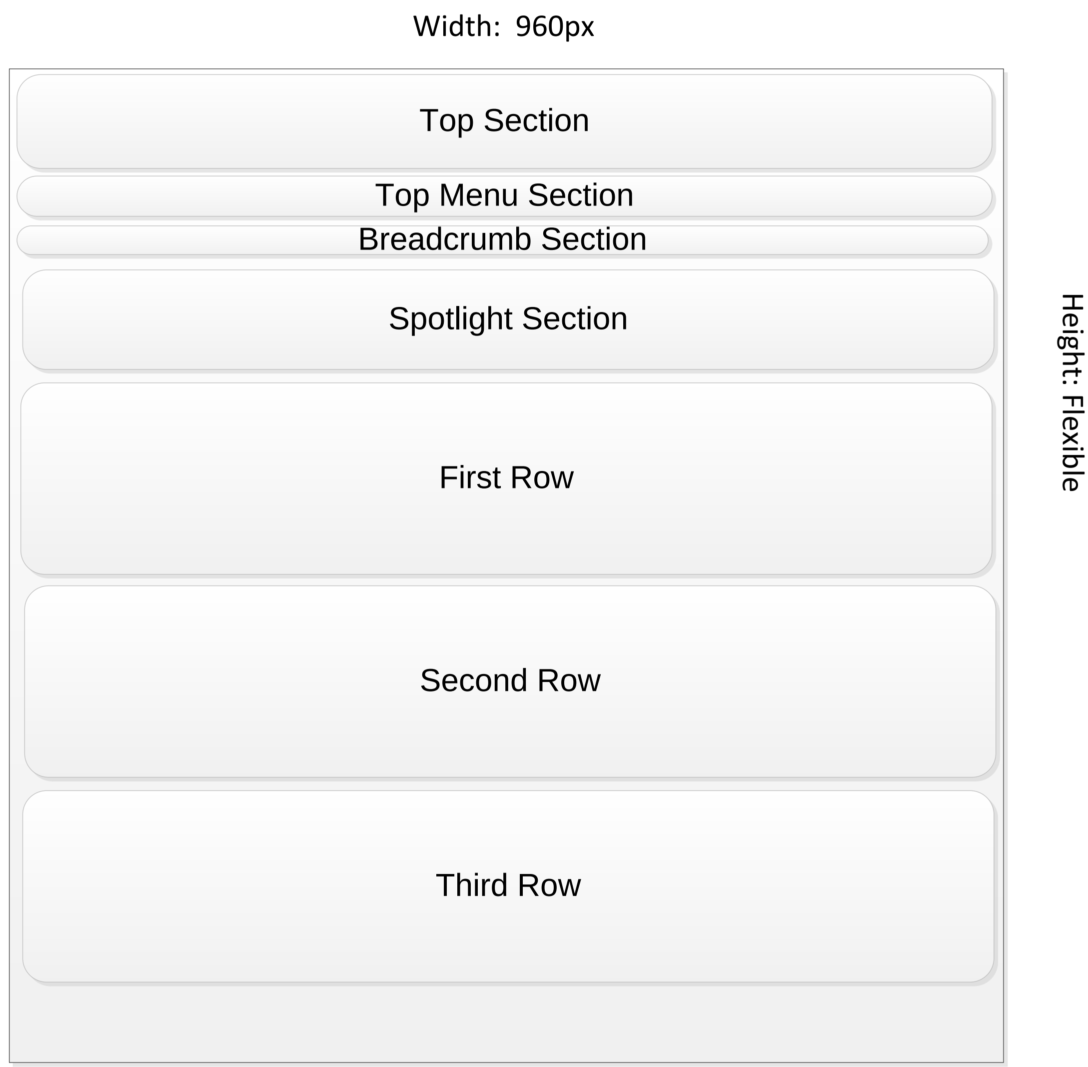

Figure 16: Master page template

The dashboard is broken into five main sections, or pages: Overview, Operations, AMHS, Engineering, and Planning. Each of these pages represents different areas or levels of the fab which allows users to view the areas that they are interested in and to drill-down from high-level data to more specific data. Each page is broken into "components" that consists of a chart, list, or a combination of both that represent a specific metric used by users. These components can be added, modified, or replaced depending on client needs. Each component includes the ability to download the raw data included in the component in an Excel spreadsheet so that data can be used by users for other purposes if needed. They may also include links that allow users to drill down to pages that contain more detailed data. In addition to the examples shown in Figures 17 and 18, the top menu and the spotlight are components. The top of each page includes a menu for navigation between the main pages and a search box that allows users to search for modules, toolsets, tools, and lots. Also available are links to the "Share Page" and "Feedback" functions as seen in Figure 19. In Figure 20, the layout of the website can be seen.

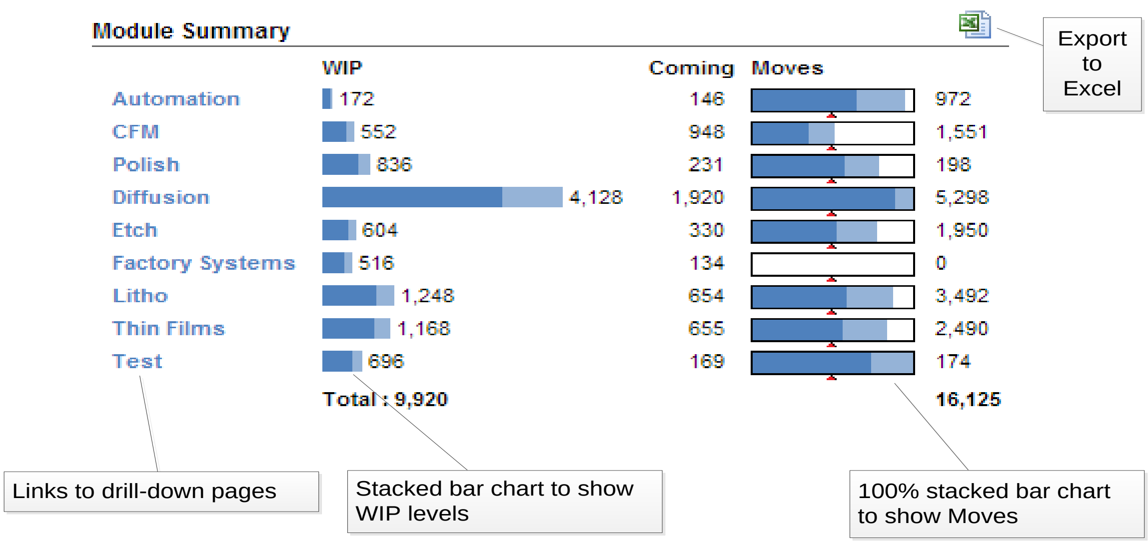

Figure 17: Module Summary component

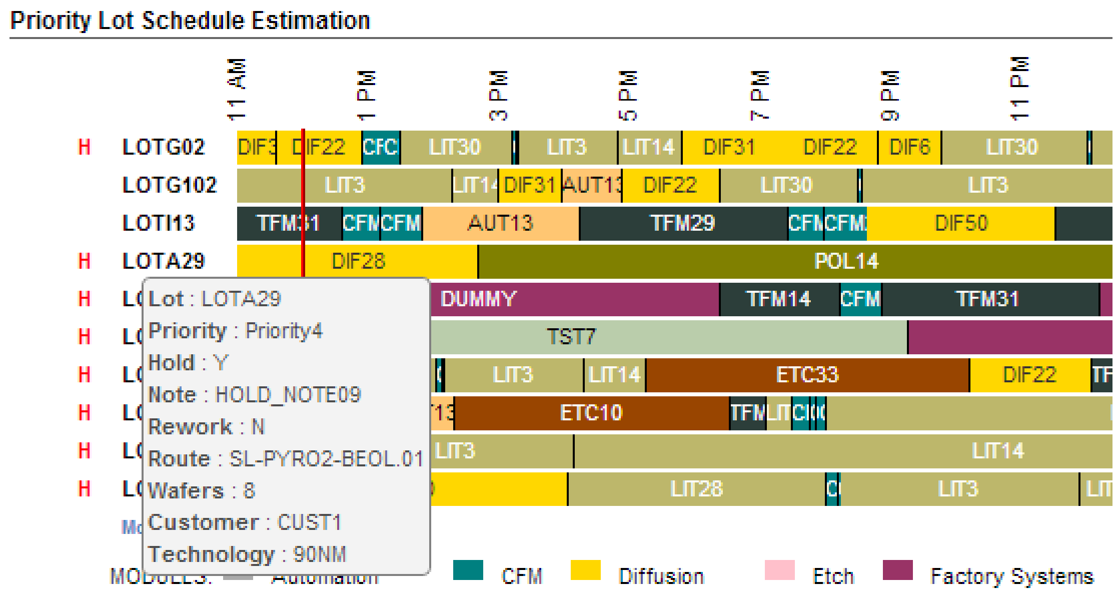

Figure 18: Snapshot of scheduling chart with hover message

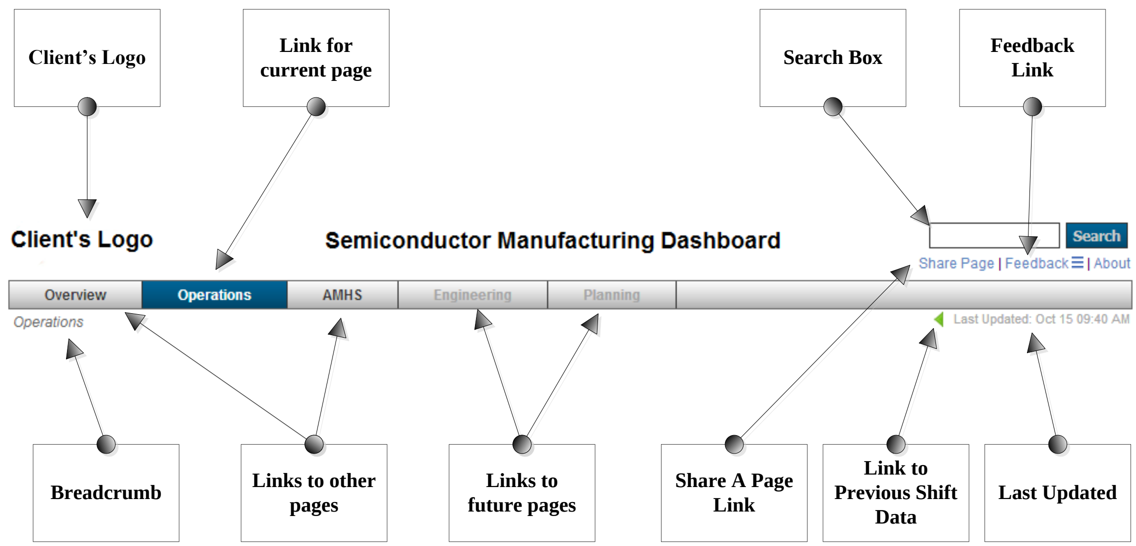

Figure 19: Snapshot of Top Menu toolbar with descriptions

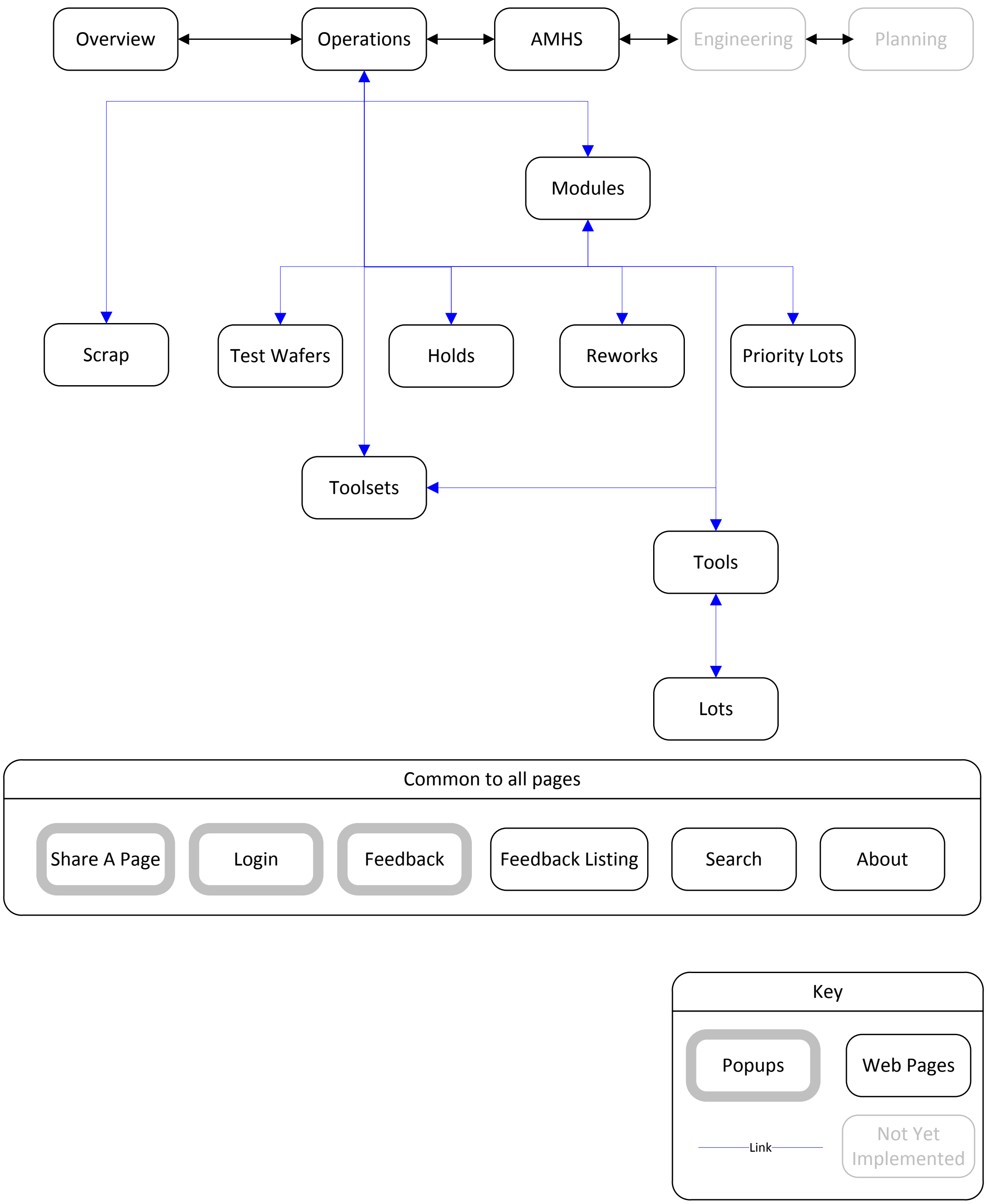

Figure 20: Dashboard website layout

Code-Behind Pages

As explained earlier, every active server page displayed actual consists of a file that displays content to the user, the front page, and a file that contains code that runs on the web server and contains the logic for what should be displayed on the page, the code behind page. The source code for the code behind pages are written in .NET C# and are responsible for populating variables used by the front page, connecting to the database and retrieving data, and error handling.

Overview Page

The Overview page consists of charts that present a high level key metrics of the fab that are used by high level management to monitor production and to help make manufacturing decisions. These metrics could include the mix of technologies that are being manufactured, actual shipments compared to planned shipments, WIP and Die Per Mask Layer (DPML) per technology actual number versus targets, on time per delivery per customer, and overall line yield versus targets. Due to the design of the application, charts and grids of various other metrics can also be added or removed depending on the requirements of the client.

Operations Page

Whereas the Overview page was designed to provide data to high level fab managers, the Operations page was designed to provide data and key metrics to module managers who need to see information specific to their modules and high level information about the overall condition of the fab for the current shift. Currently, the components on the page includes a summary of modules, toolsets, tools, priority lots, and processes as well as a listing of any line holds that may be active. Again, components can be added, removed, or edited depending on the needs of the client.

Located at the top of the page is a "spotlight" component (See Figure 21) that highlights key fab metrics that are of most interest to managers. With one glance, managers can see the overall condition of the fab and how these metrics measure up to specified targets. Each metric is within the spotlight is contain within a "button." Depending on the metric these buttons may provide hyperlinks that allow users to drill down to obtain more specific information about the metric. The data for the previous shift is also included at the bottom of each button for easy comparison between current progress and the previous shift. Current metric values are color coded. Numbers display red to indicate that current values are behind current expected values, yellow indicates that at current pace, there is a rick that shift targets will not be met, and green indicates that current values are on track to meet or exceed shift targets.

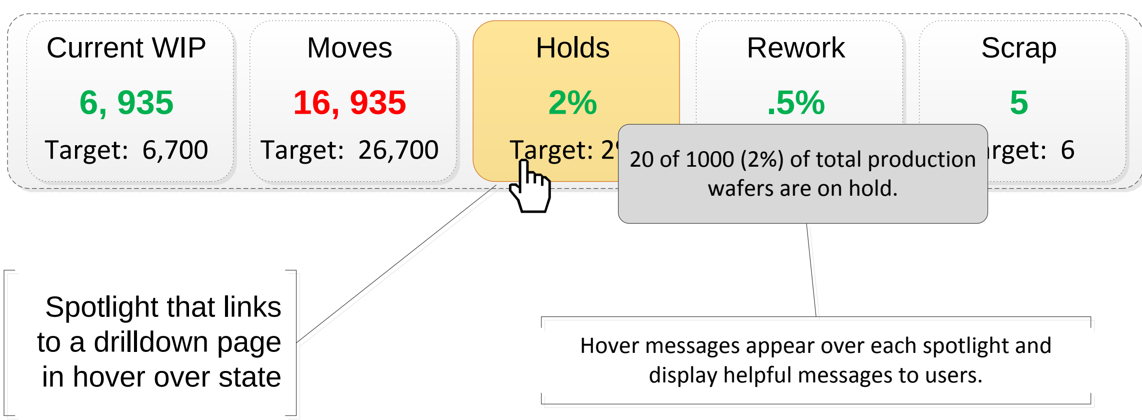

Figure 21: Spotlight Example

Automation Material Handling Systems (AMHS)

As mentioned earlier, the AMHS is the system that is responsible for automatically moving manufacturing material throughout the fab. The AMHS page is currently under development and currently on contains information about WIP storage levels. Future plans are to include:

- AMHS moves per hour
- Average Move times
- Stocker Fullness
- Number of stockers down for maintenance
- A diagram on the physical layout of the stockers and graphics to display their operational status and level of fullness

Engineering and Planning Pages

These pages are designated for future development and will include information useful to engineering and planning personnel.

Drilldown Pages

From the main pages, hyperlinks are provided in specified components that allow users to drilldown to see more detailed information about a specific item. This could include module, toolset, tool, lot type, hold, test wafer, rework, scrap or any other information deemed important.

Historical Views

The ability to navigate back and forth in history is available on the Operations and Module pages. Just below the top menu on these pages, arrows are provided to allow users to navigate back and forth between shifts. The ability to just to the current shift at any time is also provided. See Figure 22.

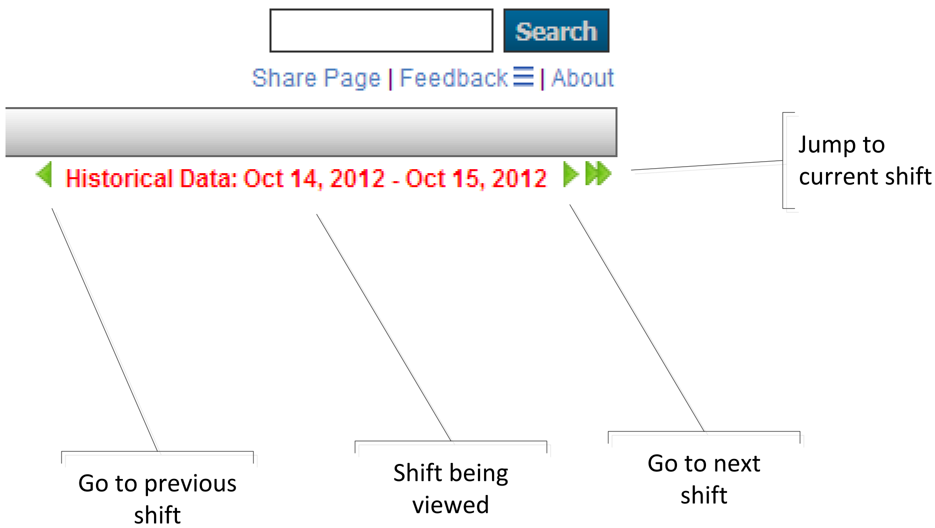

Figure 22: History navigation

Currently, only historical views of the Operations and Module pages are available although this feature is also planned for the Overview page.

Web Page Updates

Earlier the updating of data in the database was explained. One of the biggest advantages of the dashboard over traditional reports is that the data can be updated in real-time. Using AJAX, calls are made to the Oracle database to determine if data has been updated in the database every 30 seconds. If the data has been updated, another call is made to the database that retrieves the necessary data and updates the data displayed on the page (See Figure 23). Since Update Panels and AJAX are used, only the areas on the page that need to be updated are updated without needing to refresh the entire page.

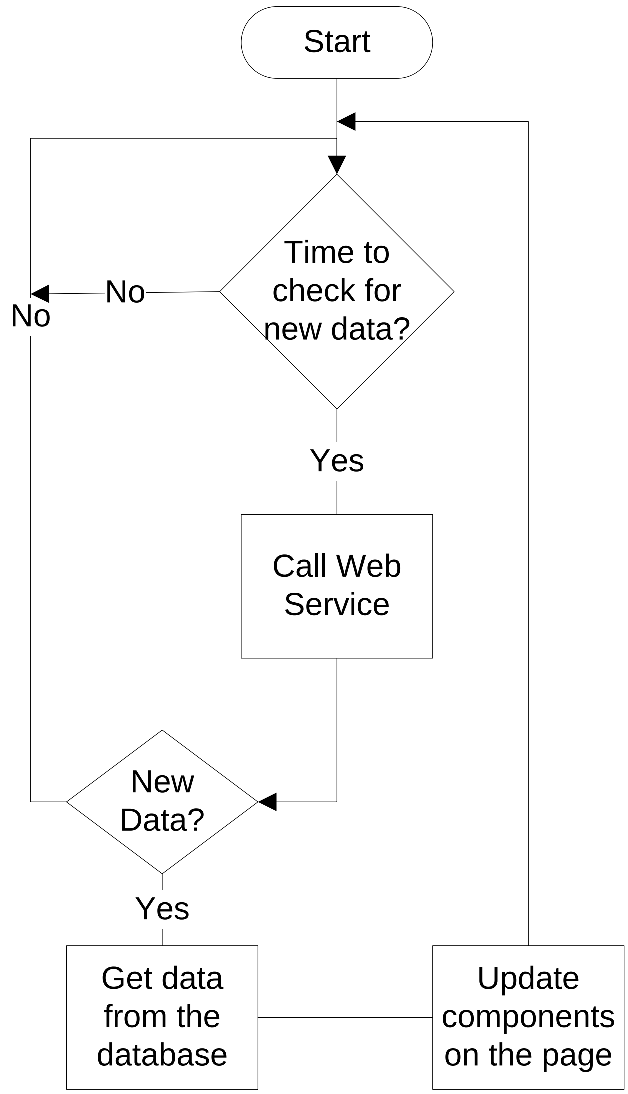

Figure 23: Flow chart for updating data on a page

FUTURE DEVELOPMENT

Although the SMD has been in production for over 6 months at a client's site, this application is constantly evolving as improvements are made and features are added. The primary method for gathering requirements for future development is from use of the Feedback application discussed earlier. As use of the SMD application increases, more and more feedbacks are being submitted from users requesting features and new views on the data. These feedbacks are gathered and discussed within the company in biweekly meetings. When examining feedbacks, each feedback is examined based on the type of feedback it is and whether the item requested is feasible, useful, and fits in the SMD. Since currently the development is being financed by a client, all requests are submitted to the client representative for approval.

There are three types of requests made: bug fixes, feature enhancements, and additional features. Feature enhancements are anything that could change or improve an existing page or component. Additional features could include a new page, component, hover message, etc. Bug fix request are examined to determine the severity of the bug and prioritized appropriately.

Feature enhancements and addition features are first examined to determine if they are feasible. A couple reasons that a feature may not be feasible are that the technology to develop the feature is not supported in the SMD or that the data is not available or accessible by the SMD. Feature enhancements and additions are also determined if they fit within the SMD application. Some reasons that a feature may not be considered for the SMD are that the data is just a list of static data that could obtained elsewhere, that the feature does not fit in the existing sections of the dashboard, or, that the data is historical in nature that might be better obtained through existing reporting methods. Requests are determined to be useful if that feature provides necessary functionality and/or data to the user.

If the enhancement or additional feature is feasible and fits within the SMD, it is then decided whether the feature is useful. Initially, the requestor of the feature is determined. If the feature does not conflict with other features, the request is immediately added to a list for future development. If conflicts exist, this request is submitted to a representative at the client representative for a decision. If there is a request that the company determines should be included in future SMD releases but the current client does not approve, this feature may still be added to a list of items that are to be added to a separate version of the SMD to be sold to other clients. Once an item is approved, it is determined what data is necessary to support the feature.

From those requests, the next tasks are to add a tab for maintenance that links to a Gantt chart that displays preventive maintenance issues, when they are due, and how many are overdue; add the AMHS page as discussed earlier; and, change chart rendering from using server-side charting components and code to a charting API that renders the charts within the web browser using JQuery. There are still plans to move this application from the ASP.NET Web Forms programming model to the ASP.NET MVC model.

LESSONS LEARNED

Developing a product of this type can be difficult. Customers have a vague idea of what they want but so do not know exactly what is possible. So, a prototype is quickly developed to show the possibilities. Once the prototype is presented, the customer has numerous ideas about what information should be included and how it should be shown. The temptation for the management of small companies is to develop the product as quickly as possible while the customer is excited about the product in order to please the customer. Soon the prototype is now the foundation for the application and little time is given to set expectations, obtain requirements, and design the software and system architectures before writing code. Although a quick initial roll out of the product was accomplished, there have been numerous updates and changes that could have been avoided if more time was given to the setting customer expectations, acquiring of requirements, and to software and architectural design.

Much time was spent on database changes and code rewrites when it was determined that the way the dashboard displayed WIP was not the way that the customer wanted to see it and that the moves being shown were correct but should be broken into different categories that were more useful to the customer. These changes took numerous man hours to complete and, because little time was spent on software design, each change had to be made on every page where this information was shown and in five different components used to show the data. Since these components contain many of the same items, currently, development is underway to combine these five components into one component.

Initially, simple calculations were accomplished in the SQL statements included on the server-side pages. As customer requirements were gathered, changes were needed.

The customer also decided that only they could deploy updates. Therefore, it was determined that all calculations should be removed from the pages and views should be used to handle the calculations and provide the data to the pages. This would allow calculations changes to be made in the database without the need to deploy new code. This led to the creation of approximately 74 tables and 182 views. It quickly became confusing to manage all of these tables and views. It was difficult to determine which view contained what data and, if changes were made to a view or a table, what effect would that change have on other views. This led to the development of the database table and view prefixes shown in Table 3. This change also took numerous hours to change the database tables and views, to test these new items, and to make the code changes necessary for the pages to use these new views. Much of this work could probably have been avoided with some initial requirements gathering and planning.

At this point in development, it is going to be time consuming to change the SMD from using the ASP.NET Web Forms model to the ASP.NET MVC model. This is made more difficult by the never ending requests for additional improvements. This change would not be noticeable to the customer but could improve future development of the project by allowing unit testing and removing the reliance on ASP.NET controls. This move could also make it easier to move to other manufacturing industries because the data models are separated from the view pages in MVC.

In conclusion, the SMD is a reliable and valuable application for monitoring the manufacturing process. After the initial prototype, time should have been taken to gather more requirements with regards to how the different groups within the FAB currently view the data and what would be useful for them. Examining source code deployment responsibilities and requirements would also have been useful. Based on those requirements and expectations, design decisions could have been made that could have taken time spent on rewriting code and making database changes and enhancing the SMD or developing new products.

References

1. Microstrategy Inc. (n.d.). MicroStrategy Business Intelligence. "Business Dashboards Gallery." Retrieved October 25, 2012, from http://www.microstrategy.com/software/business-intelligence/dashboards-and-scorecards/gallery/.
2. Centigon Solutions. (n.d.). Centigon Solutions. "Sales Dashboard." Retrieved October 25, 2012, from http://centigonsolutions.com/components/components_demosales.htm.
3. QlikView. (n.d.). QlikView. "Introducing the QlikView Business Discovery Platform." Retrieved October 25, 2012, from http://www.qlikview.com/us/explore/products/overview.
4. Quirk, Michael, Serda, Julian. Semiconductor Manufacturing Technology Instructor's Manual. Page 26. Prentice Hall. November 2000.
5. HP. (May 17, 2012). HP. "WorkStream: HP AllianceONE Partner Program." Retrieved October 15, 2012, from http://h21007.www2.hp.com/portal/site/dspp/menuitem.5179cc2b5bf6406ac6713f8da973a801/?productId=10770.
6. Hattori, Ryuhichiroh. (July 8, 2008). "IBM High Tech MES Solution." Retrieved October 15, 2012, from http://www-903.ibm.com/kr/event/download/200807_366_industry/s366_b02.pdf.
7. Applied Materials. (n.d.). Applied Materials. "Automation Software." Retrieved October 15, 2012, from http://www.appliedmaterials.com/services-software/automation-software.
8. Applied Materials. (n.d.). Applied Materials. "Factory Performance. APF RTD/APF Reporter." Retrieved October 15, 2012, from http://www.appliedmaterials.com/services-software/library/apf-rtd-apf-reporter.
9. Applied Materials. (n.d.). Applied Materials. "Factory Performance. Activity Manager." Retrieved October 15, 2012, from http://www.appliedmaterials.com/services-software/library/applied-activity-manager.
10. Hanselman, Scott. (November 10, 2011). ASP.NET. "Choosing the Right Programming Model." Retrieved October 1, 2012, from http://www.asp.net/web-forms/videos/how-do-i/choosing-the-right-programming-model.
11. Final Phase Systems. (n.d.). Final Phase Systems. "Factory Dashboard." Retrieved October 25, 2012, from http://www.finalphasesystems.com/index.php?option=com_content&view=article&id=26&Itemid=34.