Photolithography Management Web Application

Atmel Corporation

By

Cache Hamm
Jeremy Powell

June 23rd, 2006

Executive Summary

The goal of this project was to advance a photolithography product management application to an advanced beta stage. This application was to be browser based, with advanced search and editing functionality. The databases created to serve this application would keep a record of the current status of all products, as well as a historical record. Upon a search request, the application should display a filtered grid of results, which could then be selected, edited, and the history viewed.

The project was implemented using ASP.NET as the application foundation. Visual Basic .NET (VB.NET) was used for server-side execution and process requests, and the database was developed and executed on MS SQL Server 2005. The application was developed in Visual Web Developer, a web development tool available from Microsoft.

As the project evolved, some of the original requirements were changed or removed. Our team has implemented all requested functionality. The application supports the requested search features, mass and single editing, and a current and historical database. Concurrency checking was a non-requested feature the team had wanted to implement, unfortunately the time constraints did not permit this. Graphing functionality was intended to be implemented only if time allowed, which it did not.

Overall the team is pleased with the final product. All goals were met, and in some cases, exceeded.

CONTENTS

Abstract

Atmel Corporation requests a replacement software system that would allow the photolithography process to be managed more easily. This new tool will be a web browser based application and include functionality not available in the current program. Its purpose is to allow Atmel's engineers to quickly search the company product list and adjust their properties. The core features will include the following:

- A new, normalized database
- Database search functionality for all data fields
- Individual product editing
- Mass editing functionality for multiple products
- A new and improved Graphical User Interface(GUI)
- Email notification to all relevant departments with each database change.
- A historical database logging all changes made.

Introduction

- Background

Founded in 1984, Atmel Corporation is one of the largest semiconductor manufacturers in the world. They produce a broad range of products, featured in everything from iPods to television remotes. After surviving the bursting of technology stocks at the turn of the century, Atmel is currently facing a changed market with stronger competition. Because of this new market environment, Atmel is undergoing a company-wide effort to increase efficiency. This new efficiency model will enable Atmel to gain a

competitive advantage in the marketplace. As part of this effort, it has asked this team to write a more effective, user friendly replacement for its current photolithography management program.

- Goal

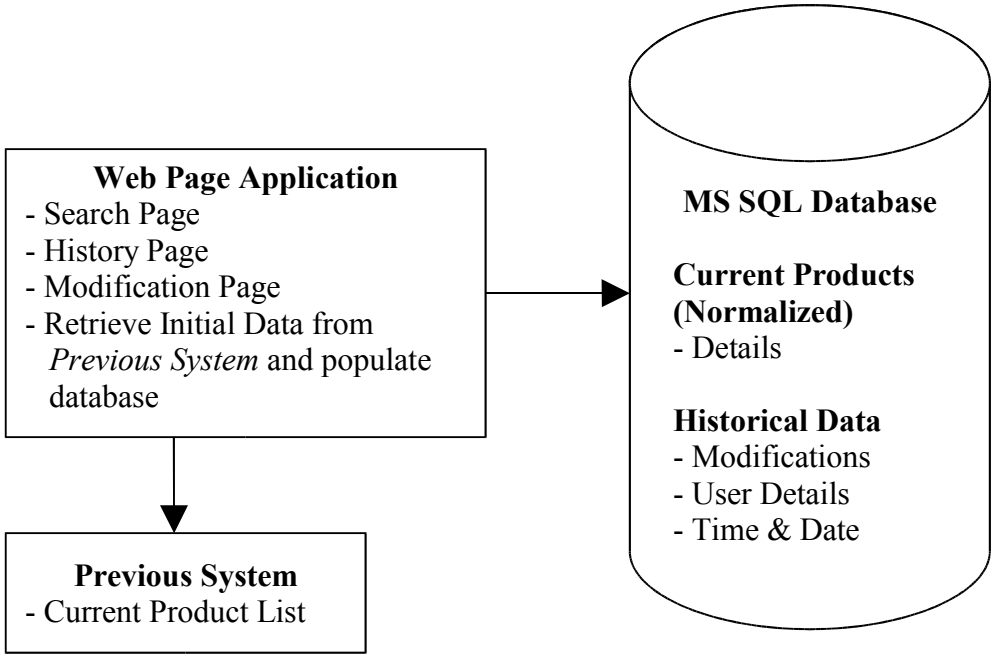
The goal of this project is to develop a browser based photolithography product management application. This application will include advanced search and editing functionality. A normalized database would hold current products and product parameters. A historical, non-normalized database would log any changes made to the normalized database. Search results will be presented in an easily readable grid format, which can then be selected, edited, and viewed historically.

Requirements and Specifications

- Functional Requirements
    - Multiple Search Fields
        - Currently, Atmel's program only allows searching for a product based on one data field of the product. Atmel wishes to extend the search functionality to provide its users to search for a product with respect to any or all data fields.
        - "Wildcard" searching must be supported, allowing the user to search a partial string and get all products containing that string returned.
    - Editing
        - The users must have the ability to update product values used in the photolithography process.
        - The editing engine must include the ability to edit multiple items at once.
        - The dropdown menus must enable the user to change the current property to a new value, or to increase (or decrease) the property by a fixed offset or percentage
    - After a user has changed some value(s), an email notification must be sent to all affected departments with the appropriate information.

- Nonfunctional Requirements
    - The application must provide the user with better functionality and a more intuitive user interface. The application must be browser based, allowing employees to access the program anywhere in the world via the Atmel website
    - Search textboxes for every product parameter must be included on the search page
    - The product database must be normalized and efficient.
    - The historical database must store comments, time, date, changes, and the user name of the editor.
    - The editing engine must be complemented by textboxes and drop down menus, for each editable field.

Below is a diagram regarding the interaction between the application, the database, and the previous system.

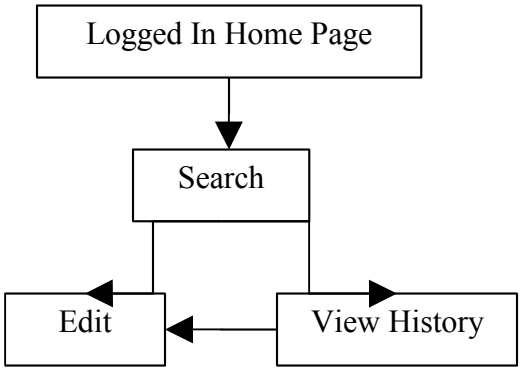


Design

- ## Overview

After constructing the flow diagram below, we were able to determine that the final product would contain three dynamic web pages; Search, Edit and History.

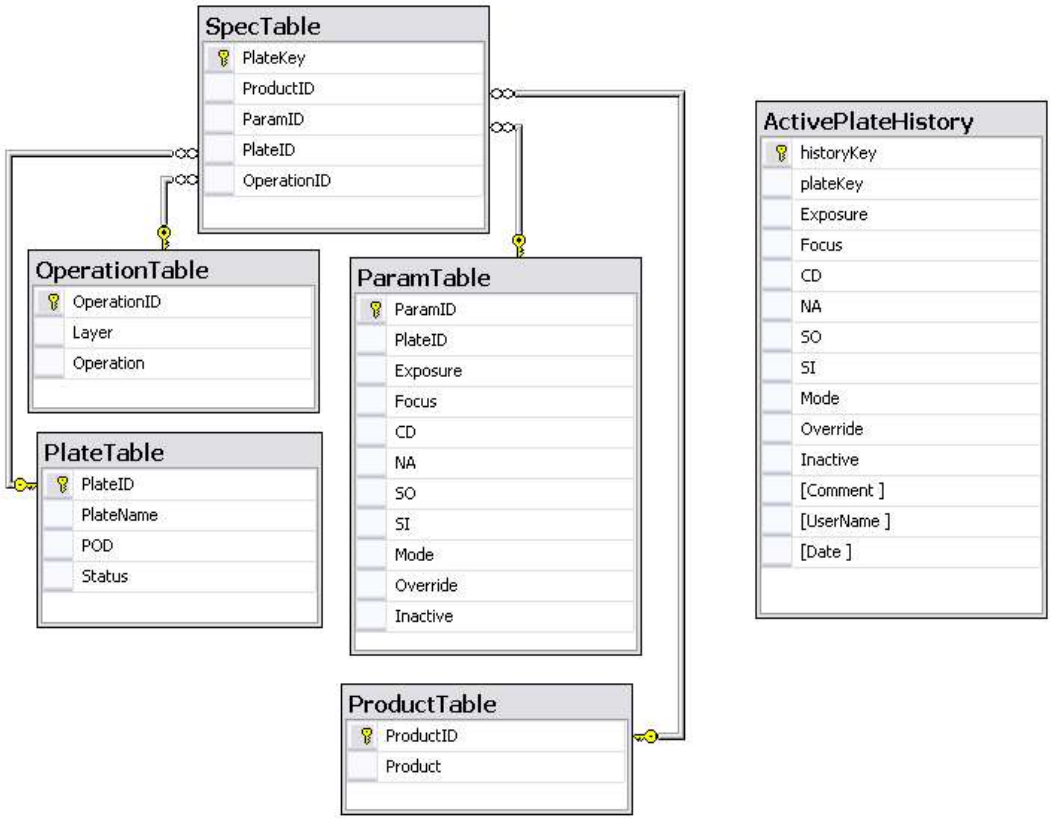
- ## Design Components



  - Search
    - The search page will query the database based on the user's filter parameters in the text boxes. It will then populate a grid with the results returned by the query.
    - Check Boxes will populate the far right column of the grid, allowing the user to select certain products.
    - "Edit Selected" and "View History" buttons will be provided on the bottom of the page. These buttons will send the selected products to the edit or view history pages.
  - View History
    - View History will query the historical database based on the products sent by the search page. The results will be displayed in a grid.

- Edit
  - The edit page is sent product values from the search page, which it also displays in a grid. The user may then choose to change the parameters for the displayed products using the options populating the drop down menus.

  - The drop down lists will be populated with '_', '+', and '%' signs or a letter depending on the type of the column.
    - When "_" is selected, the database column will be set to the value of the textbox.
    - When "+" is selected, the program will add the database value to the textbox value and store the result back into the database.
    - When "%" is selected, the program will increase the database value by the percentage in the textbox and store the result back into the database.



- Current Platelist
  - The current platelist must be normalized to maximize efficiency
  - The current platelist will have five tables: PlateTable, OperationTable, ProductTable, ParamTable, and SpecTable, the primary table containing the foreign keys.
  - Each table will hold several of the 15 parameters for each product.
- Historical Table
  - The historical table platelist will not be normalized, it will be a single table
  - The historical database will log all 15 product parameters, as well as a date, user name, and comment.

Implementation

Atmel suggested the application be developed using the .NET framework, version 2.0. The team took this advice and decided to develop in Visual Web Developer, a Microsoft .NET environment. Visual Web Developer combines coding and a graphical user interface to provide debugging, code formatting, prewritten code, and database management services. Overall, .NET and Visual Web Developer have proven to be a wise choice, as it has given the team increased design flexibility and shortened the development life cycle.

There were three sides to the development of this project; client side programming, server side programming, and the databases. The client side programming was primarily written in HTML and ASP.NET, and debugged in Firefox and Internet Explorer. The server side programming was written in VB.NET. The database and tables were created using MS SQL Server Management Express and SQL scripts.

- First Iteration : Database Development

The application must access two database tables; a table containing the current state of all products, and a table containing historical information. The initial database design called for a simple n x 15 table for both the historical and current platelist tables. Later, it was decided that the current platelist table should be normalized and split into five different tables.

In normalizing the database, a VB.NET script was written to import the platelist.txt file into an MS SQL database table. The table was then split into the five tables described above using MS SQL statements. Foreign and primary keys for each table were set using the MS SQL Server Management Express.

The historical table was created using the n x 15 current platelist table as a template and adding the additional columns for comment, date, and user name.

- Second Iteration : Search

The search page was the first page developed for the application. It was designed cosmetically first, creating a user friendly interface of textboxes and a search button. An ASP.NET gridview was created below the textboxes. When the user clicked the "search" button, an MS SQL SELECT statement was called to query the database using the textbox values as parameters. The results of this query were configured to populate the gridview.

The default textbox values were set to a percentage symbol ('%'), the "wildcard" value for SQL. This ensured that if the user wanted to filter only one parameter, the SQL

statement would still use wildcards to query the other columns, and the gridview would be correctly filtered by the one parameter.

A new column was created on the far right of the gridview and populated with checkboxes. This enabled the user to check the results of their query that they were interested in. Finally, two buttons were created below the gridview: "View History" and "Edit Selected"

- Third Iteration : Editing

The edit page was by far the most challenging aspect of this project. ASP.NET has built in functionality for individual editing, however, Atmel had requested *mass* editing functionality. Unfortunately, there was no easy way to expand the prewritten individual editing and a new system had to be designed from the ground up.

First, a gridview was populated using the selected products passed from the search page. Drop down menus and textboxes were added above the gridview, with each textbox corresponding to the column below it.

A simple SQL statement was written to update a single row in the gridview using the textboxes as parameters. Once this was shown to be working correctly, VB code was added to support the dropdown menu functionality.

Finally, a loop was created which ran through each row of the database, calling the SQL update statement each time. This allowed multiple rows to be changed at once, completing the mass update functionality Atmel requested.

- Final Iteration : Historical View Page and Email

Compared to the editing SQL statements, the historical database was very easy to accomplish. A page was created with two gridviews, one above the other. The top grid was populated using the selected products passed from the search page. Hyperlinks labeled “history” were adding to a new column on the far right. When a hyperlink was clicked, the lower gridview queried the historical database and was populated with this history of the corresponding product.

Emailing was simple to implement into the system. Code was added on the edit page to send an email to all affect Atmel departments when a product was changed. This email contains the identifying fields of the updated product and user comments.

Results and Conclusion

All goals have been met. A functioning application has been created with all requested functionality. There still remains a few out of scope requirements, but the application as a whole is complete. The current platelist database is normalized and searchable. The search results are editable with email sent to all necessary departments. A historical database is updated with every change and is viewable on request.

Future Plans

The next step for this application is the addition of the remaining features necessary for completion. Concurrency checks and historical database graphing must be implemented. The installation and set up of the system is well documented, but could be automated with the implementation of scripts. It is the team’s recommendation that the original text file database is retired from use and the normalized database used in its

place. If both databases are to coexist, then each some form of communication with the original text file database should be established to ensure that each database is kept current.