TECHNICAL RESEARCH REPORT

An XML-Based Approach to Integrating Semiconductor Process Information

by Jing Chen, Raymond A. Adomaitis

TR 2002-45

An XML-based Approach to Integrating Semiconductor Process Information

Jing Chen and Raymond A. Adomaitis
Department of Chemical Engineering
University of Maryland
College Park, MD 20742

Abstract

This paper discusses the development of a Java and XML-based semiconductor process information system for experimental data archiving, processing, analysis, presentation and simulation. The three-tier system architecture consists of a web-based GUI interface module, a business-handling module and a data services module. The user-friendly web-based GUI interface enables users to look up experimental results according to operating conditions, or to determine material and process gas physical properties using a physical property estimator, or to query and retrieve experimental data and run client-side simulations in the MATLAB environment. Simulation results can be compared to experimental data to validate process models or used to optimize the operating conditions. The business-handling module interprets user's request to determine which information is required and how that information is extracted and synthesized from the back-end database. After this information is obtained, it is delivered to the users in an appropriate format (XML/HTML).

Keywords: XML, Java, Information Systems, Semiconductor Processes, CVD.

1. Introduction

In the semiconductor industry, the manufacturers have an overwhelming amount of process and metrology data to maintain on the ICs that they manufacture. However, different data formats limit the data exchange among applications, which motivates the development of computational methods for efficiently accessing and sharing the data, extracting information from the data, and making use of the information. To enable interchange of these data, an industry consortium group, the Pinnacles Group, designed an industry-specific SGML (Standard Generalized Markup Language) markup language. Their idea was that the data wrapped by industry-specific SGML would enable intelligent applications not only to display semiconductor data sheets as readable documents but also to drive ULSI design processes. For example, engineers can access a manufacturer's website to download viewable data on particular integrated circuits or to model those circuits in various combinations in a Java applet [1]. This approach's problem was that

SGML is so complicated that almost no software has ever implemented it fully. Programs that implemented or relied on different subsets of SGML were often incompatible with each other [2].

To effectively integrate process and metrology data with distributed process simulation methods, our approach is to store CVD process data information in XML format for web applications and distribution. XML (extended Markup Language) is a descendant of SGML, but has only 20 percent of SGML's complexity [3]. XML is a simple, well-documented, straightforward data format. XML documents are text and can be read with any tool that can read a text file. The most common applications of XML today involve the storage and transmission of information for use by different software applications and systems. New technologies and frameworks (such as web services) depend heavily on XML content to communicate and negotiate between dissimilar applications [2].

This paper discusses the ongoing development of a Java and XML-based semiconductor process information system for experimental data archiving, processing, analysis, presentation and simulation, which is based on a flexible, scalable and manageable three-tier architecture. By separating presentation, business logic, and data layers into their own components, we can change the implementation of each layer without affecting the other layers. A physical property estimator for determining properties of gases is also developed. The data extracted from backend database is delivered as an XML document for further computation and distribution.

2. System Architecture Overview

The integrated semiconductor process information system is based on a three-tier architecture design, which provides us with a scalable, reusable, manageable and flexible solution and greater security structure to share data locally and remotely.

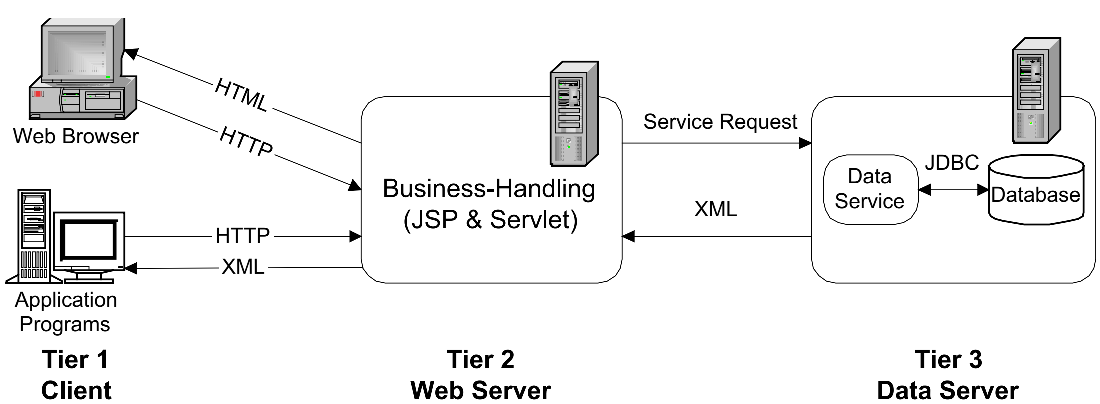

Fig 2.1. Integrated Semiconductor Process Information System Architecture

As shown in Fig.2.1, the information system consists of three modules: 1) a web-based GUI interface module, 2) a business-handling module, and 3) a data services module. The web-based interactive GUI interface enables users access the system services easily, querying and retrieving experimental data (such as photos of wafers and deposition thickness), looking up species physical properties and so on. The business-handling module interprets user's request from the front end to determine what information is required and how that information should be extracted or generated from the data service module. After this information is obtained, it is synthesized into a suitable format for the user. The data service module hosts various databases and some computation modules for different services. XML facilitates handling of various data sources/types.

3. Design and Implementation

3.1 Web-based GUI Interface Module

The web-based GUI interface may provide following services to its users: experimental data query, thermo-physical property estimation and reaction simulation. The communication protocol between client and server is HTTP.

3.1.1 Experimental Data Query

Users select the experimental conditions in the query request forms and submit them through the Internet. The business-handling module in the middle layer pre-processes the requests and invokes the corresponding data service in the backend. The resulting data would be returned in XML format. If no experimental data are available to the user at the specified conditions, current available experimental conditions may be returned. The server side business logic process this XML wrapped data and render the results in an appropriate HTML to the users for display.

3.1.2 Physical Property Estimation

The interface provides the methods for determining properties of non-polar and polar gases and organic and inorganic gases with emphasis on supporting semiconductor process gases such as silane, tungsten hexafluoride and trimethylgallium. The thermo-physical properties include viscosity, thermal conductivity, diffusivity, heat capacity, molecular weight and density of pure gases and mixtures at ideal gas state. Properties of 62 pure gases, binary or even multi-component gas mixtures consisting of those pure gases in various units may be obtained through the interface. [4]

3.1.3 Reaction Simulation

To reduce network traffic and user waiting time, the overall system architecture was designed to support client-side simulations, especially in the MATLAB simulation environment. Developing simulations and applications on the client-side offers users the functionality to compare simulated results with experimental data to validate the model or to use simulated results to optimize experiments. This application also gives users more flexibility to manipulate experimental data. Different users could use data for different purposes, i.e., plotting and analyzing experimental results, or doing simulation work and so on. To retrieve data needed from the backend database, users specify the operation conditions with selected reactions and submit it. The business-handling module takes the request and invokes the corresponding data service model in backend. The data service module takes the request, retrieving experimental data set from experimental database, species physical properties from physical property database, and reaction information from reactions database, then combines all information together and wraps the data in XML format. This package is sent back to middle layer for further distribution. After users get the packaged data from business model, they will parse this XML file and get all the information they need for computation or other usage.

3.2 Business-Handling Module

The business-handling module, hosted on the web server, is implemented with Java Servlet and JSP (Java Server Pages) technology. Java Servlet is supported directly or by a plug-in on virtually every major web server. Java Servlet is easier to use, more efficient, powerful and portable, safer and cheaper than traditional CGI (Common Gateway Interface). Servlet also has an extensive infrastructure for automatically parsing and decoding HTML form data, reading and setting HTTP headers, and many other such high-level utilities [5].

The business-handling module interprets user's request to determine what information is required, how that information should be extracted or generated from the data service module, and what kind of document format should be present to its final customers. It has two layers: the business logic layer and the server-side presentation layer. All clients communicate with the web server through HTTP protocols to invoke the business logic layer’s Servlets, which in turn calls the data services module to request the corresponding services. The data service layer takes the query and access various databases in the backend by JDBC (Java Database Connectivity) technology. The resulting native or manipulated data are sent back to the middle layer in XML document formant. In the presentation layer, by using JSP and Java Servlet technology this XML document can be delivered to the client in the appropriate markup format (XML/HTML) for further applications/distributions or display in web browser.

3.3 Data Services Module

The data service module hosts some data manipulation sub-modules and various databases supporting these services. The data service layer is developed with JAXP (Java API for XML Processing) and JDBC technologies. JAXP has a standard set of Java APIs to deal with XML objects. JDBC is a standard SQL (Standard Query Language) database access application interface. Together, JAXP and JDBC provide an infrastructure for building applications using XML and SQL [3]. To support the services offered by the data services module, three databases are used to store experimental data, reaction information, species thermo-physical data, and process equipment data, which could be queried through JDBC. The experimental database includes equipment/facilities data (such as wafer size), experimental conditions (such as temperature, pressure, concentration, flow rate, etc.), and experimental results (such as images of wafer, metrology data, deposited layer thickness or weight). The reaction information database has data about reactants and products, reaction equations, reaction constants, activation energy and reaction order for both gas reactions and surface reactions. The physical property database contains viscosity, thermal conductivity, diffusivity, heat capacity, molecular weight and density of pure gases, which currently only support gases or species for semiconductor process use. Whenever properties of a binary or multi-component gas mixture are requested, functions in the data access service layer are called to generate them under ideal gas state on the fly. The requested data results would be wrapped in an XML format and then send back to the business-handling module for further applications or distributions. RDBMS (Relational Database Management System) are used to ensure robustness, integrity, consistency, and availability of data.

4. Conclusions

An XML-based semiconductor process information system is under development for experimental data archiving, processing, analysis, presentation and simulation. Based on a three-tier architecture, it provides a scalable, reusable, manageable and flexible solution and greater security structure to share data locally and remotely. The system is being developed using Java and XML technology. Data representation in XML format is platform and vendor independent, which makes data from a variety of sources available to be used to drive a variety of distributed applications.

5. Acknowledgement

The authors acknowledge the support of the National Science Foundation thought grant ECS-0082381.

References

1. Jon Bosak, http://www.ibiblio.org/pub/sun-info/standards/xml/why/xmlapps.htm, 1997.
2. Elliotte Rusty Harold, W.Scott Means, XML in a nutshell, O'REILLY, 2002.
3. Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto, XML and Java, Developing Web Applications, 1999.
4. Junghwan Kim, http://www.glue.umd.edu/~jvkim/jpp/, 2002.
5. Marty Hall, More Servlets and JavaServer Pages, Prentice Hall PTR, 2002.