

WEB-BASED TOOLS IN WAREHOUSE DESIGN

*Douglas A. Bodner, T. Govindaraj, Edgar E. Blanco,
Marc Goetschalckx, Leon F. McGinnis and Gunter P. Sharp*

Keck Virtual Factory Lab
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0205 U.S.A.

Abstract. The world-wide web is emerging as a powerful platform for distributing information and supporting interaction within collaborative groups. In an era of global supply chains, flatter organizations, and distributed decision-making, web-based technologies enable faster and better warehouse design and also greatly improve the process for individuals learning about warehousing. This paper highlights some key web-based technologies in the context of warehouse design, and presents an interactive web-based environment that integrates a design database, pick area design heuristics and cost models. The environment is in use as an experimental testbed for education and training.

Introduction

The world-wide web is emerging as a powerful platform for distributing information and supporting interaction within collaborative groups. In an era of global supply chains, flatter organizations and distributed decision-making, web-based technologies are creating new opportunities to improve the state-of-the-art in facility design and analysis. Here, we focus on warehousing and discuss improvements in the processes of designing warehouses and of educating people about warehousing.

As a domain, warehousing is increasingly competitive and complex. Consider the business trends having an impact on warehousing and distribution. Greater product customization and niche marketing are creating a proliferation of stock-keeping units (SKUs). Customer expectations for shortened lead-times are increasing the pressure on warehouses to perform. Warehouses are being called on to handle value-added transactions, such as specialized labeling, kitting and assembly. Third party logistics providers are seeing financial opportunity in providing warehouse services to manufacturers and distributors. Global competition is causing business to focus on warehousing and distribution as an area of competitive advantage. Together, these factors point toward greater difficulty and importance being attached to the creation of good warehouse designs.

Today's business environment may be more complex, but today's computational tools are commensurately more powerful. The question is how to leverage them to

improve the process of designing facilities. In the remainder of this paper, we describe several new technologies that can be applied to the problem of designing facilities, with a concentration on warehouses. After discussing the specifics of the warehouse design problem, we present a prototype web-based design environment that can be used to make design decisions and perform analysis for a subset of problems in warehouse design. Our goal in this paper is to present the technologies that are available and discuss how they can be applied to improve the state-of-the-art in warehouse design.

Web Technologies

Within the past five years, a variety of new technologies have revolutionized the way that computational models can be conceptualized and deployed. Here, we briefly describe relevant tools and technologies and how they might be leveraged in the context of warehouse design.

Web Browsers

Perhaps the most obvious technology is the *web browser* itself. The two major browsers are Microsoft® Internet Explorer and Netscape® Navigator. Fundamentally, the browser is an implementation of the client-server paradigm of computing. The browser serves as the client, communicating with a server that might be located anywhere. The server manages data, handles user sessions, performs analysis, and reports results back to the user. In terms of facilities design, a browser allows the user to enter data and queries and to interact with a model on the server. The model has the algorithms and logic needed to support design activities, and these can be made transparent to the user. For specialized applications, a browser may require a plug-in (e.g., to view a text document, a spreadsheet, or even a 3-D visualization of a warehouse).

Web Servers

To host a web site, a server needs to have *web server* software installed. This type of software allows users to access files on the server computer using standard internet protocols (e.g., hypertext transfer protocol, or HTTP).

Web servers also provide the capability for secure transactions.

Application Servers

The web server handles basic functionality for web pages. For more advanced functionality, a web site requires an *application server*. The application server handles interaction with databases, and it provides a robust architecture to support complex computing applications on the server. In addition, application servers provide powerful development environments. For advanced web applications, the application server functions as the application engine, as it manages data and user sessions, performs computations either internally or by invoking other applications, and returns results to the user via dynamically created web pages. For example, an application server might call a simulation package or an optimization package to perform some analysis needed for a warehouse design.

Scripting and Programming Languages

Programming languages continue to be important in the development of web-based applications. The introduction of the Java™ language has created a platform-independent way to run programs. Hence, Java is quite popular as a language. Java™ provides functionality for applets, which are programs that can be downloaded from a web site and run on a user's computer. In addition to Java™, there are a number of scripting languages that are used for programs that run on the server. These can be used to implement that logic needed for facilities design models.

Databases

While not a new technology *per se*, databases serve an increasingly important role in web-based applications due their data-intensive nature. Relational database systems are fairly common. Object-oriented database systems are an emerging technology.

Databases are used to manage data about a particular system under study. For example, a database may contain the past year's customer order information for a warehouse, or it may contain cost information for storage and material handling equipment, or it may contain labor productivity standards. Databases also are used to manage collaborative user sessions, to store designs in progress, and to archive completed designs. Types of databases range from desktop applications (e.g., Microsoft® Access) to industrial-strength applications (e.g., Oracle®).

Middleware

Most web-based applications are distributed in the sense that they incorporate a number of different software components that interact with one another. Moreover, these software components might reside on a number of different computers. Hence, there is a great need for software and standard protocols to manage these

distributed applications so that they are integrated seamlessly. This type of software often is called *middleware*.

For example, consider a user who wishes to use a model on a server to configure a storage system. The user must enter parameters about the system under study. In all likelihood, the model is not structured in a manner amenable to the way in which a user would like to enter data. A standard protocol is needed to take user input data and enter it into the model in the appropriate places. In other words, this standard protocol translates between the user interface and the model. Likewise, a standard protocol is needed to transform the output of the model to a dynamically created web page and transmit it over the web. A variety of software implementations exist for these types of protocols.

Visualization

In facilities design, there is a strong need for visualization, in the form of charts, block layouts, and even 3-D renderings. Certainly, desktop applications have powerful visualization capabilities, ranging from the chart and graph capabilities of spreadsheets, to 2-D drawing packages, to 3-D computer-aided design applications. Each of these types of visualization are useful in facilities design. Charts and graphs can explain patterns in customer orders or inventory levels. Two dimensional drawing packages such as Visio™ can support block layout design. Computer-aided design packages such as AutoCad® support 3-D renderings and are the bridge between the facilities designer and the architectural and engineering team that builds the warehouse.

Web-based visualization technology is beginning to develop these kinds of capabilities. One emerging technology is Virtual Reality Modeling Language (VRML). VRML provides a set of generic 3-D objects that can be used to create 3-D "worlds." VRML can be used to visualize a warehouse design and allow a user to "walk through" a virtual warehouse facility. VRML viewers are implemented as plug-ins to web browsers.

One current limitation of visualization technologies is that there is limited capability for "drag and drop" functionality over the web.

Examples

This section has discussed web technologies and how they can be used in the context of computational tools for warehouse design. Table 1 provides examples of the technologies described in this section. Today, many of these technologies are freely available.

The Warehouse Design Problem

Before discussing specifics of how web technologies can impact warehouse design, we first discuss the warehouse design problem.

Table 1. Examples of web technologies

Technologies	Examples
Web browsers	Microsoft® Internet Explorer, Netscape® Navigator, OmniWeb™
Web servers	Apache™, Jigsaw™, Microsoft® Internet Information System, Java Web Server™
Application servers	GemStone™, Sapphire/Web™, SilverStream™, WebLogic™, WebObjects®, WebSphere™
Programming and scripting languages	Perl, PHP, WebScript, ASP (Active Server Pages™), ColdFusion™, JSP (Java Server Pages™)
Databases	DB2™, FrontBase®, Informix™, Oracle®, MySQL™, PostgreSQL™, SQL Server™, Sybase™, Versant™
Communications protocols and interfaces	CORBA (Common Object Request Broker Architecture), DCOM, OMG Interface Definition Language
Visualization	VRML (CosmoPlayer™, Cortona™), Java 3D™

Design Issues

Warehouse design can be characterized as a set of design decisions intended to result in a facility that will meet certain requirements. These requirements may include meeting a specified throughput or storage capacity, meeting a specified level of customer service, or satisfying a budgetary constraint. The particular design decisions involve such things as determining labor requirements and automation levels, selecting material handling and storage technologies, sizing and configuring storage systems, determining layouts and material flows, etc. Of course, individual designers might have preferences that affect how they design a warehouse.

The design process usually takes input data in the form of customer order history, inventory history and an SKU database. An order history database may have tens or hundreds of thousands of records. Hence, computational tools are important for the designer. The designer typically has access to cost information, equipment data, and standards data (e.g., labor productivity). Finally, if the design involves use of an existing facility, the designer has information about the facility itself, which in general constrains the design.

The Forward-Reserve Problem

Here, we focus on a subset of the warehouse design problem, namely the specification of the forward pick area and reserve storage area within a warehouse. The forward pick area is set aside to enable rapid picking of products from stock to fill orders. Intuitively, a warehouse manager would like to facilitate this type of rapid picking for those SKUs that are ordered most frequently (i.e., the "fast

movers"). Typically, products are stored in cases or broken cases, and the storage technology is selected to facilitate rapid picking. For example, a flow rack system might be used whereby an operator can pick items and then place them on a belt conveyor to be sent to an order sorting and accumulation area.

Reserve storage, on the other hand, is used to store large quantities of products, typically in pallets. Reserve storage relies on storage efficiency – how to maximize use of a given space to store large volumes of products. Thus, a storage technology for this area will be selected on the basis of how efficiently it can store products. Examples include blockstacking (i.e., storage of stacked pallets on the floor) and rack storage (i.e., storage of pallets within a rack structure). SKUs stored in the forward pick area might also be stored in bulk in reserve storage. When the inventory for SKUs in the forward pick area becomes low, those SKUs are replenished from inventory in the reserve.

Figure 1 illustrates a warehouse with a forward pick area and a reserve storage area. The arrows represent a material flow pattern for items stored in the forward pick area. In this pattern, incoming products are received, then stored in pallets in reserve storage. As orders are received for these products, the products are picked from the forward pick area and then are sent to the order sortation and accumulation area. From there, they are shipped. As inventory levels in the forward area decrease, products are picked from the reserve storage area to replenish the forward pick area. Of course, other material flow patterns are possible and even likely. For example, products not stored in the forward area (e.g., "slow movers") are picked directly from reserve storage for outbound orders.

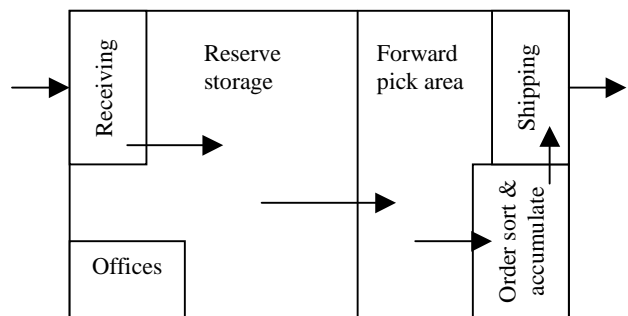


Figure 1. Warehouse with a forward pick area

The forward reserve problem is described in more detail by Frazelle *et al.* (1994). Obviously, there are trade-offs and cost considerations involved. Making the forward pick area large, for example, improves order-filling efficiency and decreases labor costs for order picking up to a point, since order picking costs are lower for items in the forward area as opposed to those in reserve storage. However, it decreases overall storage efficiency, since the space for reserve storage will be decreased. In addition, as more products are stored in the forward area, the labor cost

of replenishing these items from reserve storage increases. Another important consideration involves storage equipment, in terms of cost and relative efficiency for order picking. The following costs are relevant to the forward-reserve problem:

- Capital cost of equipment,
- Labor cost associated with replenishing the forward area from reserve storage, and
- Labor cost associated with order picking.

In designing a forward pick area versus a reserve storage area, critical design decisions include the following:

- Determine the size of the forward pick area;
- Determine the set of SKUs to be stored in the forward pick area;
- Determine the amount of each SKU to be stored in the forward pick area; and
- Determine the storage technologies to be used.

The goal is to make these decisions so as to increase efficiency and lower cost. In the next section, we discuss computational tools to aid with these decisions.

A Web-Based Design Environment

As an example of how web-based tools can be integrated into a design environment, we now discuss a prototype application developed at the Keck Virtual Factory Lab at Georgia Tech. This prototype serves several purposes. First, as a demonstration platform, it shows how a web-based application can be used in the context of warehouse design. Second, it serves as a tool to educate students and train practitioners in the area of warehouse design. Finally, it demonstrates the advantages and disadvantages of different web technologies.

Requirements

To be effective, such a web-based application must conform to several requirements. The first major requirement is that it support a *thin-client interface*. Here, a thin-client is defined as follows:

- A web browser (with Java™ enabled);
- Minimal software to be installed on the client. Typically, this software will consist of easily (and freely) available plug-ins.

A related requirement is that the client be supported on multiple platforms, ranging from Windows® 95/98, to Windows NT®, to Mac OS, to Unix (e.g., Solaris™ by Sun Microsystems, IRIX® by SGI, or Linux™).

Finally, the client should provide the user the means to view available data and make design decisions, and the server should provide functions to evaluate designs in terms of cost and provide feedback to the user.

Use Cases and System Architecture

One fundamental question in creating a computational application is how will the user interact with it. The idea is

to create a specification for the application that will support the functionality needed by the user. One way to accomplish this is through *use cases*. A use case simply describes how a user interacts with a computer application to perform a task. By specifying a set of use cases, it is more likely that the application will be designed to support the intended functionality. Further information about use cases is available in [Fowler and Scott, 2000].

The development of use cases is important in any software system. Below are examples of use cases that are important in the context of a computational application for the forward-reserve problem.

- View demand and order pick data for all SKUs.
- View storage and inventory data for all SKUs (e.g., maximum stackability, days of inventory kept on hand, etc.).
- View cost data for all storage technologies available.
- View technical data for all storage technologies available (e.g., maximum lane depths, lane clearances, number of unit loads that can be stacked, etc.).
- Specify the area dedicated to the forward pick area.
- Partition SKUs into those that are stored in the forward pick area vs. those that are not.
- Allocate space in the forward pick area to selected SKUs.
- Specify and configure storage technologies (e.g., specify lane depths for blockstacking).

Data for products, storage technologies and costs are captured in databases on the server. These databases also are used to manage user sessions. For example, if a user does not make all design decisions in one session, the history of his or her session is kept so that the session can be resumed the next time that the user accesses the application. Heuristics for cost and efficiency calculations are implemented on the server and are invoked by the user to evaluate a design. Results from the heuristics are reported back to the user via dynamically created web pages. Figure 2 shows the architecture for the application.

Implementation

Originally, the warehouse design environment was implemented in Perl as a prototype. Perl is a scripting language for web applications. The databases were implemented using PostgreSQL, an object-relational database system. The application uses the Sun web server, and it is hosted on a Solaris™ workstation.

To improve robustness and aid maintainability, the design environment is being re-implemented using an application server. WebObjects® from Apple has been chosen, primarily because it is based on mature technology originally developed by NeXT. It also includes a powerful development environment. The heuristics will be implemented in Java™, while the databases will remain in PostgreSQL.

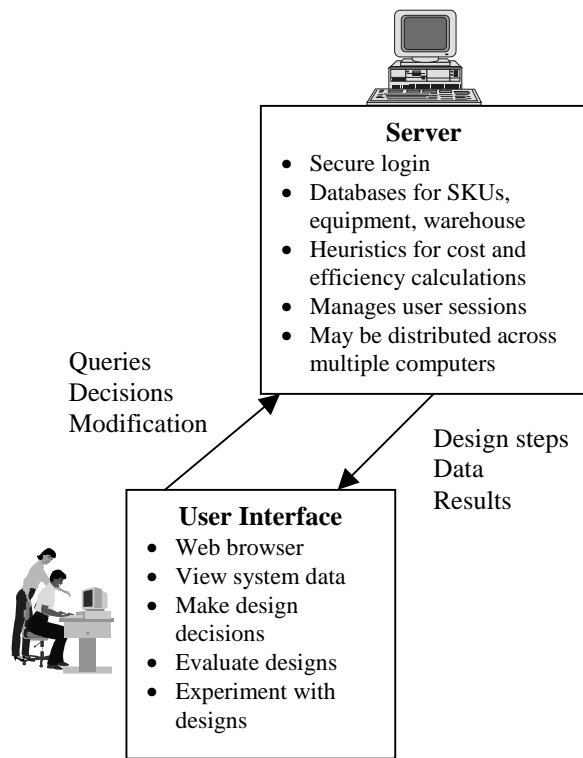


Figure 2. Computational architecture

The design environment is available via the website of the Keck Virtual Factory Lab at Georgia Tech (<http://factory.isye.gatech.edu>).

Functionality of the Design Environment

In this section, we outline the basic functionality and uses of the warehouse design environment. The user needs only a Java™-enabled web browser.

Functionality

When a user logs into the design environment, he or she is asked to provide a user ID and a password. This allows the system to provide the user with access to any previous design sessions that might be in progress.

Once the user is logged in, he or she may select from a number of scenarios. At present, these include warehouses and distribution centers for computers, apparel and groceries. The data for these scenarios resides on the server (SKU data, equipment data, cost data, etc.). Once the user selects a scenario, he or she is provided with a set of tables from the databases on the server. These tables contain product demand data, storage equipment technical data, costs for labor and equipment, throughput estimates, etc.

Next, the user is presented with two sets of design decisions:

- Select and configure the storage technology for the reserve area, and

- Configure the forward pick area.

We assume that the storage technology is already selected for the forward pick area, namely gravity flow racks with picking handled via carts. The options for storage technology in the reserve area include blockstacking, single-deep rack storage and double-deep rack storage. The user can determine a storage technology for each SKU. The user can configure the storage technology by specifying a lane depth. For the rack storage alternatives, the lane depth is pre-defined, but for blockstacking, the lane depth is open to manipulation.

In configuring the forward area, the user has several decisions to make. The user is given a total amount of area dedicated to reserve and forward storage. The first decision is to select what percentage of that area is dedicated to the forward area vs. the reserve area. Next, the user selects which SKUs to assign to the forward area and what percentage of the forward area each should comprise.

Finally, the user costs his or her design. The cost heuristics are based on the work of Frazelle *et al.* (1994). In particular, this work uses the concept of an economic assignment quantity (EAQ) to rank SKUs with respect to whether or not they should be stored in the forward pick area. The EAQ is calculated for each SKU and depends only on the demand for that SKU and the number of times it is requested. The costs calculated by the application include the capital cost of equipment, the replenishment cost and the order picking cost for a particular design. Based on these cost estimates, the user can return to the design and make changes to see if improvement is achieved. The user can also log out during the middle of a design, saving it for later modification, or the user can close a design and archive it on the server. Figure 3 shows a screen capture of the web page for the design application.



Figure 3. The forward-reserve design application

Discussion and Future Work

This type of design environment can be used in a number of ways. One way is to educate students about the concepts involved in the forward-reserve problem. In this

use, the design environment functions as an experimental platform, so that students can learn about the impact of various decisions on costs when specifying a forward pick area. Such an experimental platform has the potential to motivate students to learn more about the theory underlying the model and cost heuristics, through the concept of active learning.

This type of design environment has potential application in practice. For example, design teams can use a web-based design environment to collaborate from remote locations. An engineer or consultant might wish to perform some analysis on the warehouse floor. Being able to access data and models over the web has important benefits in that it can save time and effort. Also, if sensitive data are stored on the server, the data can be secured there, rather than being stored on a personal computer or laptop, where the data might be compromised.

Future work involves using this design environment as a lab exercise in the curriculum at Georgia Tech and other partner universities. In addition, we are collaborating with a number of industry partners in a larger research program to create and deploy better methodologies and computational tools for warehouse design. Part of this work involves design visualization. Advances from the visualization work will be incorporated into the forward-reserve design application.

Conclusion

This paper has discussed the use of web-based tools for designing warehouse facilities and has presented an example of such tools. We anticipate that such tools will become more widespread as usage of the world-wide web increases and as web technologies mature. Because of these new technologies, there are important opportunities to improve the state-of-the-art with respect to facilities design. Our future work focuses on leveraging these opportunities in research, curriculum development and industrial practice.

Further Reading

- Fowler, M. and K. Scott. *UML Distilled, Second Edition*, Reading, MA: Addison-Wesley, 2000.
- Francis, R. L., L. F. McGinnis and J. A. White. *Facilities Layout and Location: An Analytic Approach*, 2nd Edition, Englewood Cliffs, NJ: Prentice Hall, 1992.
- Frazelle, E. H. *World-Class Warehousing*, Atlanta, GA: Logistics Resources International, Inc., 1996.
- Frazelle, E. H., S. T. Hackman, U. Passy and L. K. Platzman. "The Forward-Reserve Problem," in *Optimization in Industry 2*, T. A. Ciriani and R. C. Leachman (eds.), New York: John Wiley & Sons, 1994.
- Goetschalckx, M. and L. F. McGinnis. "Designing Design Tools for Material Flow Systems," *Computers and Industrial Engineering*, Vol. 17, No. 1-4, pp. 265-269.

Sharp, G. P. and C. S. Yoon. "A Structured Procedure for Order Pick System Analysis and Design," *IIE Transactions*, Vol. 28, pp. 379-389.

White, J. A. and H. D. Kinney. "Storage and Warehousing," in *Handbook of Industrial Engineering*, G. Salvendy (ed.), New York: John Wiley & Sons, 1982.

Acknowledgments

The authors would like to acknowledge funding from the W. M. Keck Foundation, which has made possible the activities and facilities of the Keck Virtual Factory Lab. Funding for this work has also been received from the National Science Foundation under grant DUE 9950301.

Biographical Sketches

Douglas A. Bodner is a research engineer in the School of Industrial and Systems at the Georgia Institute of Technology, where he manages the Keck Virtual Factory Lab. He is a member of IIE, INFORMS and IEEE.

Edgar E. Blanco is a Ph.D. student in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. His research interests focus on modeling of combined production-distribution systems.

T. Govindaraj is an associate professor in the School of Industrial and Systems at the Georgia Institute of Technology, with interests in human-machine systems, information engineering, and manufacturing systems. He is a member of AAAS, ACM, and IEEE.

Marc Goetschalckx is an associate professor in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. His interests include design of logistics systems, facilities and supply chain systems. He is a vice president of IIE and a member of INFORMS.

Leon F. McGinnis is a professor in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. His interests focus on manufacturing logistics systems, applied computation and applied operations research. He is a Fellow of IIE.

Gunter P. Sharp is an associate professor in the School of Industrial and Systems Engineering at the Georgia Institute of Technology. His interests include storage/retrieval and material flow systems and economic decision analysis. He is a member of INFORMS and WERC.