

WOOD - Web-based Object-oriented Desktop

Cheukyin, Chan
Department of Computer Science and Engineering
Chinese University of Hong Kong
Shatin, Hong Kong
cychan@cse.cuhk.edu.hk

ABSTRACT

In this poster, we describe WOOD (Web-based Object-oriented Desktop) which is a web-based personalization information system. It provides various types of application components to help users gather and manipulate information over the Internet effectively. Components, with different capabilities ranging from fetching from search engines to getting latest stock quote or even to clip part of a web pages, are available to the users. Users can add and layout those components in their personal page with WOOD as if they were working in the local window operating system. Since WOOD is designed to be open and extensible, third party developers can easily develop their own components and plug into WOOD. This poster briefly describes the architecture and the prototype implementation of WOOD.

Keywords

Information Retrieval, Personalization, Web applications, User Interface

1. INTRODUCTION

Let's think about the following scenario. You go online in the morning. You would like to read the news headlines in the CNN and The New York Times web sites. Besides, you would like to check your stock portfolio and bank accounts at your online financial services provider web sites to determine whether you should change your investment strategy. After that, you would also like to check out for the weather forecast, entertainment and sport news, the updates of your research fields and your book order status at Amazon. You quickly find that you visited over more than a dozen of sites and these are just routine actions. It is just too time-consuming to go over each of them every morning. What you need is a nice page which contains all of those information.

Personalized information systems, such as [1], [2], [3] and [4], try to resolve the above problem using a component model. They provide hundreds of information components about news, stock quotes or calendar. Users are also able to layout their modules on their own pages so as to build a personal information search page. There are also researches using component model to build web applications for the users, such as [5] and [6].

WOOD is also a personalized information system which has several improvements over these systems:

- **Open and extensible** : WOOD is an open and extensible web personalization system which contains

different information components. The components in other systems are usually proprietary. Third-party developer cannot construct their own customizable components easily. With this in mind, WOOD is designed to be an object-oriented system that all major system services and components core structures are built as a set of easily accessible and reusable core libraries. Through these libraries, developers can make their own invaluable components and plug into WOOD system quickly and ready.

- **Information processing and component collaboration** : WOOD helps users to analyze and manipulate information rather than being another information displaying and gathering systems. In WOOD, some of the components contain processing logics. They take the information shared by other components and apply different processing. For example, a set of image processing components are built within WOOD. Users can now flip, rotate and even blur an image grabbed from other web site. Moreover, the components are also able to collaborate with each others. In the above example, the clipping component which clips the image will work with the flipping component to complete the task.
- **User friendly interface** : WOOD's user environment is designed with users kept in mind. It resembles closely to users' local window system environment. Each component is displayed as window just like an application on users' local computer desktops. Interactions are done through traditional forms and dialogs. Working with WOOD is more or less the same as working in their computer desktops. At the same time, unlike other systems, it does not require the user to download certain plug-ins to support the windowing capabilities. The windowing subsystem is implemented in Javascript which can be viewed by popular browsers, e.g. Internet Explorer.

2. SYSTEM ARCHITECTURE

WOOD is divided into a number of modules. They are Member module, Desktop constructor, Windowing module, Component base class and Scheduler. Figure 1 shows the high level overview of the system architecture of WOOD.

2.1 Component

Components are the core and building blocks of WOOD. They implement different functions ranging from simple weather report to a sophisticated stock quote analyzer. They have to conform to the component requirements imposed by WOOD. These requirements are based on the

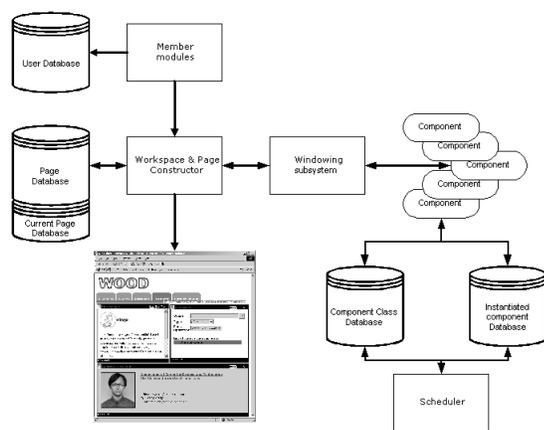


Figure 1: Architecture overview of WOOD

WOOD's mechanism in handling component Instantiation, scheduling and Presentation. These requirements are specified as a set of API that a component should implement. In practice, the Component base class implements all necessary API with default actions. All components are required to subclass from the component base class. Those APIs include:

- **Instantiation** : This API formalizes the parameter querying mechanism and provides a concrete structure in implementation for the component instantiation phase.
- **Scheduling** : This API is divided into instance based and class based scheduling. Components should implement their main routine here.
- **Presentation** : Components should output their content through the presentation API. The presentation content should be in HTML format. It can contain hyperlinks, audio, video and even applet. WOOD saves the flexibility for the developer.

Currently, we have four major types of components:

- **Aggregation** : This kind of module relies on or simply brings together other components output. Meta-search component is a typical example. It combines search results from different search engine components to compile its own list of ranked output.
- **Clipping** : Sometimes users only want a part of a web page. Clipping modules automatically help them to clip the desired section of the page on their own desktop. For instance, a user may use the Page Clipping modules to clip the headline section of a fashion magazine web site routinely.
- **Fetch and process** : Most of the components fit into this category. They fetch information from different sources and apply certain algorithms on those input. They then provide tailored information to the user.
- **Update notification** : This kind of components helps user keep an eye on the update of certain online resources. They will alarm the user for any changes detected.

2.2 Windowing subsystem

Windowing subsystem handles all the display functionalities for the components. It generates the window view in purely HTML code, Javascript and CSS. The component windows can then be viewed by popular graphical web browser without the requirement of any plug-in or Java applet support. It consists of two parts: one on the server side and one on the client side. The former handles the generation of HTML code and the latter handles the user actions and windows operations.

2.3 Scheduler

Scheduler is a standalone module that will wake up components and signal them to run their specific tasks in scheduled time. The components of WOOD are passive in nature. They only work and perform processing when requested by the system. There are, however, many tasks should be done in a timely fashion. It may take too long and make the response time unacceptable resulting in user frustration in these cases. Another reason is that there are tasks that need to be done periodically or in a specific time.

3. PROTOTYPE SYSTEM

We built the prototype implementation of WOOD with several modules developed, for instance, Wrapper, MetaAgent (metasearch components), Newspaper agents and Encyclopedia components. Figure 2 is a typical screenshot of WOOD in action.



Figure 2: WOOD in action

4. ACKNOWLEDGMENTS

This project is supported in part by a grant from the Hong Kong's Research Grant Council #CUHK 4407/99E.

5. REFERENCES

- [1] Desktop.com. <http://www.desktop.com>.
- [2] MyYahoo!. <http://my.yahoo.com>.
- [3] Octopus. <http://www.octopus.com>.
- [4] OnePage.com. <http://www.onepage.com>.
- [5] Weiquan Zhao and Jian Chen. CoOWA: A Component Oriented Web Application Model. *Proceedings of the 31st International Conference on Technology of Object-Oriented Language and System*, 1998.
- [6] Hans-W. Gellersen and Martin Gaedke, Object-Oriented Web Application Development, *IEEE Internet Computing*, Vol. 3, No. 1, January/February 1999.