

[In *Proceedings of 1st International Workshop on Web Engineering*, Brisbane, Australia, April 1998.]

Scenarios of Web-Based User Interfaces for Software Engineering Environments

Jun Han , Peninsula School of Computing and Information Technology, Monash University, McMahons Road, Frankston, Vic. 3199, Australia. jhan@monash.edu.au

Abstract

With the rapid expansion of the Web technology, the Web has been increasingly used for the user interfaces of software applications. In this paper, we present and analyse a number of scenarios in which the Web can be used for the user interfaces of software engineering environments. These scenarios focus on the integration of a new Web-based user interface with existing components of a software engineering environment through the use of the Web/Java technology, the CORBA distributed object technology and the object database technology. The scenarios presented are also applicable to the integration of Web-based user interfaces into general (legacy) software applications involving databases.

Keywords: User interface, software engineering environment, system architecture, Web, Java, CORBA, ODBMS.

1 Introduction

The user interface is an essential component of a software engineering environment (SEE). Given the wide spread use of the Web, especially as user interfaces to various software applications, it is natural to expect that the user interface of a software engineering environment should use the same technology, to give the software engineers (i.e., the users of the environment) a look-and-feel similar to the Web, while having the specific functionality of user interfaces in traditional environments.

In this paper, we look at ways to integrate the Web technology into software engineering environments as their user interface components. In contrast to developing new environments, the scenarios we present focus on the integration of a new Web-based user interface with existing components of a software engineering environments such as its object management system (OMS), process engine (PE) and other tools. Such integration scenarios are based on the Web/Java technology, the CORBA distributed object technology and the object database technology (ODBMS). In the following sections, we present four Web-based user interface (integration) scenarios for software engineering environments. For each scenario, we outline the integration architecture and analyse its characteristics.

2 The gateway/database scenario

In many cases, a software engineering environment, especially its object management system, is developed based on a database system for managing the software artifacts developed using the environment. In particular, the recent object database systems such as O2 facilitate direct connectivity to the Web, through the gateway technology (i.e., the common gateway interface -- CGI) on the Web server side [Gundavaram96]. That is, objects (or software artifacts) can be viewed and edited through the Web browser. In essence, the gateway carries out the software artifact conversion task between the Web-page format required by the browser and the

object format in the database. In the case of O2, the gateway works according to a default conversion scheme, and the scheme can be customized globally or on an object/class basis [O2Web96].

In this integration scenario (see Figure 1), the software developer uses the Web browser to view and edit software artifacts according to the format of the generated pages. In effect, the Web browser forms the user interface of the integrated software development environment. Since the gateway and the default conversion scheme are provided with the database, it simplifies the user interface development task. However, the default conversion scheme is usually data/object oriented, and therefore may not be suitable to software artifact conversion. For example, the presentation of software artifacts (e.g., programs) often requires the merge of many artifact objects (at different nesting levels) into one Web-page in a specific format. Furthermore, some parsing and unparsing scheme is required during this conversion process. In general, object/class level customization is required. While the provision of a default conversion scheme is convenient for most (data) conversion scheme, the default conversion framework may prove to be too restrictive (in terms of customization) for complicated conversions (e.g., those required by software artifacts).

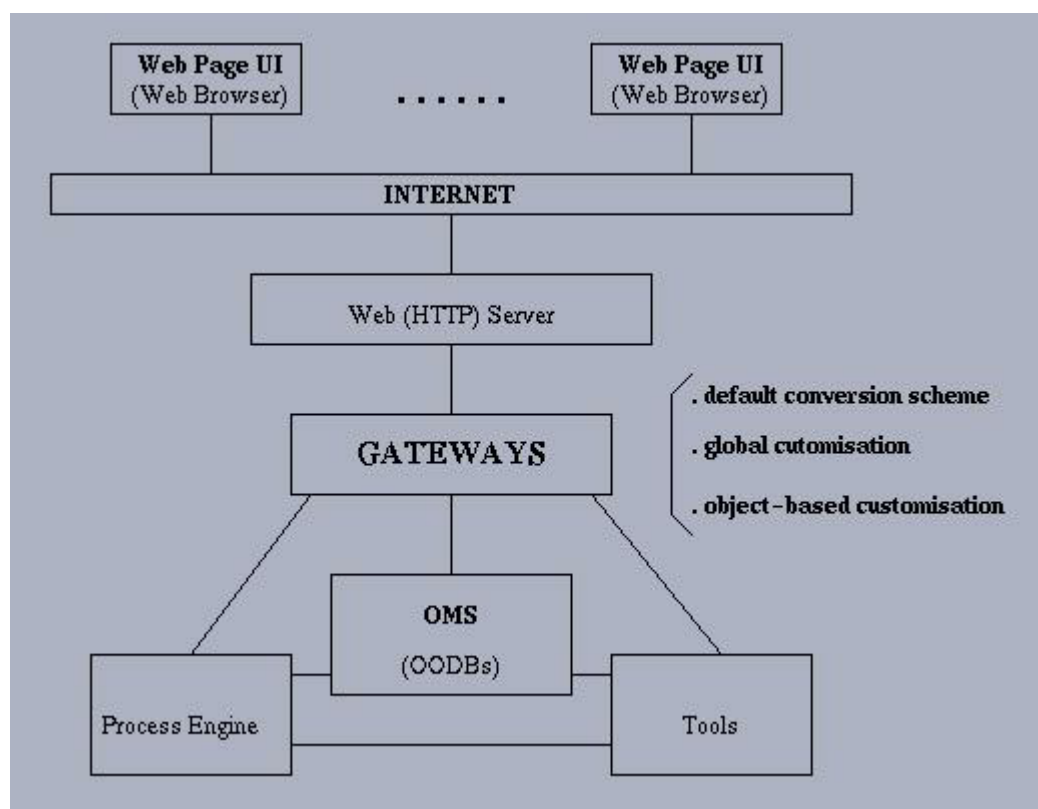


Figure 1. Integration Architecture of the Gateway/Database Scenario

It should be noted that one does not need to rely on an existing gateway. Rather, he/she may develop his/her own gateway with a specialized conversion scheme. But, much work is required. Besides, the database does not have to be an object database, as long as the gateway technology is available and possible for the integration.

3 The applet/database scenario

With the integration of the Java technology and the Web technology [Flanagan97] and that of the Java technology and the object database technology comes another scenario in which we can use the Web as the user interface for software engineering environments. This scenario assumes that there exists a Java binding with the ODBMS (such as O2 [O2Java96]) and the environment components (including the OMS) are developed surrounding the ODBMS in such a way that newly written Java applets can interact with the components (such as the process engine) and access the software artifacts in the OMS. The environment components may or may not be written in Java. In the latter case, the interaction with Java applets can be achieved through the ODBMS's

capability of being able to bind to different languages in an inter-changeable manner (as in O2). In this setting, we can develop a Web-based user interface for the environment using applets, which interact with the other environment components through the ODBMS and access software artifacts in the OMS. The execution of the applets in the Web browser provides the user interface.

In this scenario (see Figure 2), the user interface applets have to be properly structured and implemented to provide the required conversion functionality (e.g., the parsing and unparsing scheme) between the artifact objects stored in the OMS/database and the user-oriented Web-based presentation. In reality, it is more appropriate to develop/customise the environment's user interface management system (UIMS), as a Java server-side, to perform the conversion task and interact with other environment components. The user interface applets will become light-weight and communicate with the UIMS. In comparison, the applets and the UIMS in this scenario replace the gateway and related customization in the above gateway/database scenario. In this scenario, there is no pre-existing default conversion scheme, the entire conversion functionality has to be implemented in the UIMS component. As indicated above, this may require more effort, but may prove to be necessary as required by the complexity of the conversion. Compared to the development of a gateway, this applet alternative may prove to be more effective in terms of the flexibility it offers.

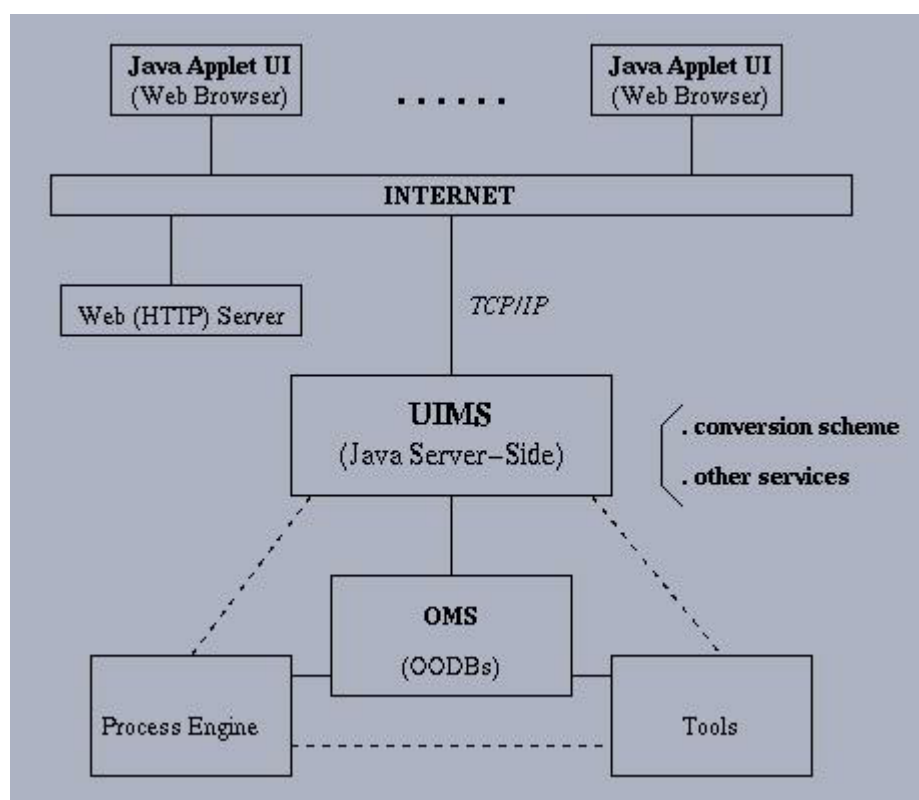


Figure 2. Integration Architecture of the Applet/Database Scenario

Again, the database in this scenario does not have to be an object database as long as the database has a Java binding, which is usually available. However, the interaction between the user interface and other environment components are much convenient to implement when using an object database.

4 The applet/CORBA scenario

In this scenario, the components of the software engineering environment are integrated through an object request broker (ORB) -- a CORBA implementation [OMG], including the OMS(s), the process engine(s) and tools. This cluster of components are not directly Web/Java-enabled. To provide a Web-based user interface for the environment, we may use the Java/applet technology as in the applet/database scenario. However, the Java server-side UIMS component is integrated to the ORB, interacts with the other environment components via the

ORB, and serves the user interface applets.

This scenario does not require that the environment components have Web/Java-enabled capability, and represents a more flexible integration scheme (see Figure 3). In fact, the environment components can be implemented in any language or database, as long as they are (or can be) integrated to the ORB. Same as in the applet/database scenario, the Java server-side UIMS has to implement the conversion between the artifact objects stored in the OMS/database and the user oriented Web-based presentation.

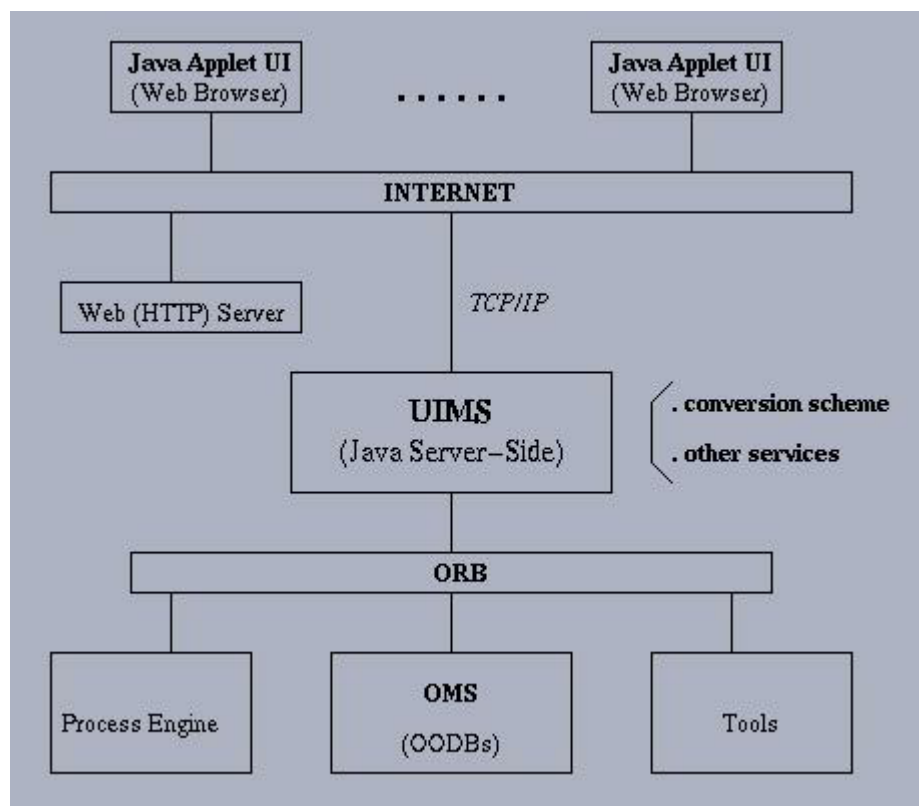


Figure 3. Integration Architecture of the Applet/CORBA Scenario

As mentioned above, the availability of CORBA provides much more flexibility for the environment components' implementation technology. At the same time, the interaction protocol between all environment components (including the Web-based interface) needs to be carefully worked out over the ORB.

5 The orblet scenario

An orblet is a Java applet that can interact with (remote) CORBA objects. This technology relies on the direct integration of Java/Web technology and the CORBA technology. Example commercial implementations of such integration are Iona's OrbixWeb [Iona96] and SunSoft's Joe. In this scenario (see Figure 4), the environment components are integrated through CORBA implementations, as in the applet/CORBA scenario. However, the UIMS component does not need to be a Java server-side, and the orblets directly interact with environment components (including the UIMS) through ORB-to-ORB connections (i.e., the orblets ORB to the components ORB), i.e., the Internet Inter-ORB Protocol (IIOP) [OMG].

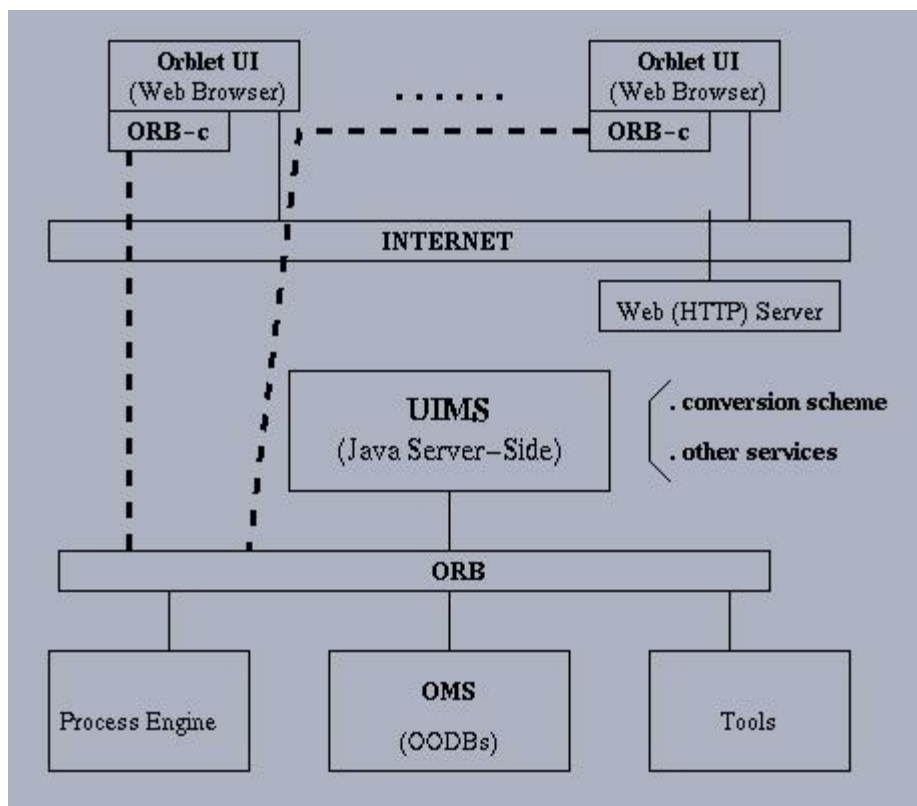


Figure 4. Integration Architecture of the Orblet Scenario

This scenario provides the similar capability and flexibility to the applet/CORBA scenario, if not greater. In fact, certain environment components such as the UIMS may be directed integrated into the orblets ORB in certain situations. This scenario requires the availability of the orblet technology, but does not require the UIMS to be a Java server-side.

6 Summary and further work

In this paper, we have presented four scenarios for providing Web-based user interfaces for software engineering environments. While providing a commonly aware user interface, these integration scenarios also adds the *distribution* capability to the environments. They assume the availability of different technologies and have their own distinctive characteristics in terms of flexibility and integration effort. We are currently carrying out some of these integration experiments and others are planned, to gain further insights into them. At a more general level, these experiences and insights are also applicable to the integration of Web-based user interfaces into general (legacy) software applications involving databases.

References

- [Flanagan97] D. Flanagan. *Java in a Nutshell: A Desktop Quick Reference, second edition*. O'Reilly & Associates, Sebastopol, USA, 1997.
- [Gundavaram96] S. Gundavaram. *CGI Programming on the World Wide Web*. O'Reilly & Associates, Cambridge, USA, 1996.
- [Iona96] Iona Technologies. *OrbixWeb White Paper*. <URL: <http://www.iona.com/>>, 1996.
- [O2Java96] O2 Technology. *Java O2 Binding: A Sneak Preview*. Versailles, France, 1996.

[O2Web96] O2 Technology. *O2Web User Manual*. Versailles, France, January 1996.

[OMG] Object Management Group. *OMG Technical Library*. <URL: <http://www.omg.org/library.htm>>.