Process*Web* - Process Support for the World Wide Web

R. Mark Greenwood, Brian Warboys

Informatics Process Group (IPG) Department of Computer Science
University of Manchester M13 9PL UK
email: markg,brian@cs.man.ac.uk

Abstract. This paper reports on a small experiment, which although relatively simple, raises important issues about Process Model User Interfaces. It outlines an approach to the exploitation of the World Wide Web by Process-centred Software Support Environments. It concludes that there is great advantage to be gained from designing process systems so that they are more reactive and rather less prescriptive than the current generation of such systems.

Keywords:*Process-centred Software Support Environments, World Wide Web*

Introduction

The problem area of concern to a Process-centred Software Support Environment (PSSE) is that of computer systems which provide a set of facilities to support people (primarily software engineers) and organisations (as opposed simply to individuals) concerned with all aspects of computer systems development [4]. The World Wide Web (WWW), provided via the Internet, also provides similar potential support.

This paper considers the possibility of integrating the systems to give some mutual benefit to both. It reports the early results of an experiment conducted by a third year student attached to the IPG at Manchester which has raised a number of significant issues concerning the relationship between the User Interface of a PSEE and the process programs with which it enacts.

Interacting with the two systems

Interacting with the WWW. Interaction with the WWW is essentially a personal business. One browses the HTML pages, HTML is the WWW text formatting language, through a series of personal choices. Each HTML page is independent of any other in the sense that there is, in general, no control system imposing any order on the browsing. Each page provides links to other pages which the user might investigate but there is no explicit ordering imposed. In short it provides the ultimate in truly democratic information searching.

By the use of *bookmarks* and *back* facilities it allows the user to retrace previous steps. It also supports a number of specific actions which can be performed on documents such as printing and *ftp*-ing. However the WWW provides no real context facilities, any page can be directly accessed as long as one knows its address. One result of this is that it has little or no protection with a number of highly socially undesirable results. Further there are few facilities for the dynamic updating of information, one needs to resort to the frequent re-reading of pages to see if they have changed. This is analogous to viewing the current state of a process.

It is in essence a *passive personal process engine*. The development of technologies such as Java[1] will undoubtedly mean that new collaborative and *active* facilities develop. For example the recent WWW pages provided for ICSE 18 allowed users to customise their registration forms. However it stopped short of providing the user with the ability to actually register electronically. The process provided meant that the customised form needed to be printed and then faxed(!) to the organisers. This example demonstrates both the interest and potential in using the WWW to deliver process support.

The paradigm is essentially one of any user providing a structured set of information and any other user being free to browse this information in any way desired. The User is clearly in control.

Interacting with a PSEE. We take the *early* ICL ProcessWise Integrator (PWI) [1, 2] as a fairly typical example of the UI to a PSEE. Virtually all current UIs to PSEE's support the notion of a work context. *Early PWI* supports the concept through the provision of a *Role Agenda*. Selection of a specific *Role*, which the user now wants to play, connects the user to the process or the context to perform some work. The user is then provided which a set of options, the *Action Agenda*, which is used by the enacting process program, a *Role Instance*, to inform the user of which actions can be undertaken. (We use the term *early PWI* to distinguish PWI releases 1 and 2, from release 3 and above which no longer have the specific role agenda, action agenda UI.)

Essentially the process program is telling the user what choices are available at this time and is also mediating the work of the team of which the user is a member. A selected action might cause the process program to send consequential actions to other users in the team. In this way collaborative processes are supported. In general the progression of the process is under the control of the process program, and is non-reversible. It is in essence an *active team-oriented process engine*.

Our experience with early PWI is that developers tend to produce process programs which drive the UI in a fine-grained manner. This gives a paradigm where the process program is firmly in control, prompting the user for input as required.

What the PSEE does provide is a persistent results space. This space essentially acts as a state for storing the *partial results* of the process. Much of this state being discarded when the process reaches certain points in its enactment. It is this property which allows

[1] Java is a trademark of Sun Microsystems Inc. Java is a programming language which is particularly suited to Internet programming: Java programs (applets) can be delivered using the Internet and executed by anyone with a Java-aware WWW browser. Further Java documentation is available from http://java.sun.com/allabout.html

us to bind users to users, tools to tools and tools to users in order to provide support for software development.

The question that clearly arises is 'can we get the best of both worlds?'. Can we on the one hand allow the user to essentially do what actions are desired in any order and let the process program intervene when appropriate to ensure that team objectives are met - the process program as mediator?

The Experiment and Observations

In Process*Web*, the WWW was linked to *early PWI* using the Common Gateway Interface (CGI) of WWW. This allowed WWW selections to invoke PWI process programs. The process programs could then send an arbitrary number of *virtual* HTML pages back to the WWW. These pages, because they are virtual, can only be accessed via the process program. Clearly the order in which these virtual pages can be read is also under the control of the process program. The user browses these pages in the normal WWW way and selects what action to initiate.

The results. As a result the UI of early PWI was essentially disabled. The user is presented with HTML scripts which describe the process in which the user is engaged. The user selects *actions* and the process responds to these actions. The process program no longer has a close control of the UI. Further the style of the UI can be varied to exploit whatever advances are made in WWW technology. (There are a lot more WWW developers than PSEE developers!).

A couple of small models were used to validate the PWI-WWW link. The first was a simple two person game which was an existing PWI model translated into Process*Web*. In this the process program retains close control of the virtual HTML pages. The second example took existing HTML pages, describing possible student projects, and extended them with a supporting process program. Students could browse the pages to see the projects available, and then enter into the process of negotiation with supervisors and allocation to a project. Both of these illustrate the collaborative facilities which process support can add to the WWW. The WWW provides users with a familiar interface and better browsing support than is typical from current PSEEs.

The WWW view. Current WWW technology represents the passive extreme of possible process support. If we encode a process into a number of WWW pages then the users are provided with advice and maybe examples of what happens if a certain course of action is pursued. This "process" has no idea as to whether the advice given was taken or even read. The PSEE/WWW system provides a context mechanism, giving privacy and ordering, and a page updating mechanism. This basically introduces a partially prescriptive environment, which has all manner of uses in the support of co-operative work.

The process view. Using PWI gives much more active process support[3]. Early PWI is very specific in presenting each user with their set of "appropriate" next actions. The process only progresses when specific actions are taken; the user is not given facilities

for browsing around the process. PWI does provide context giving users their view of the process, and support co-ordination between specific individuals. The PSEE/WWW system provides process support through an accepted UI, and encourages a UI-style which appears a benefit not a hindrance.

Conclusions

The experiment raises the fundamental question as to whether process programs should attempt to prescribe the ordering of the possible *actions* of a user or whether it should allow free access to any action, merely intervening to ensure team processes are not disrupted beyond some prescribed boundaries.

It also raises the issue as to whether process programs should directly drive the UI or whether they should effectively have no controlled UI but allow the user to browse the process state and only intervene when really necessary. Our taste is for the latter.

Acknowledgements. We should like to thank Bob Snowdon from ICL for giving us the original idea and Ben Yeomans, a 3rd year student at Manchester, for implementing the experiment.

References

1. R.F. Bruynooghe, R.M. Greenwood, I. Robertson, J. Sa, R.A. Snowdon, and B.C. Warboys. PADM: Towards a total process modelling system. In A. Finklestein, J. Kramer, and B. Nuseibeh, editors, *Software Process Modelling and Technology*, pages 293–334. Research Studies Press, 1994.
2. R.M. Greenwood, M.R. Guy, and D.J.K. Robinson. The use of a persistent language in the implementation of a process support system. *ICL Technical Journal*, 8(1):108–130, May 1992.
3. R.M. Greenwood, I. Robertson, R.A. Snowdon, and B.C. Warboys. Active Models in Business. In *Proceedings 5th. Conference on Business Information Technology*, pages 141–152, Manchester, U.K., 1995. Department of Business Information Technology, Manchester Metropolitan University, Aytoun Site, Manchester M1 3GH U.K.
4. R.A. Snowdon and B.C. Warboys. An introduction to process-centred environments. In A. Finklestein, J. Kramer, and B. Nuseibeh, editors, *Software Process Modelling and Technology*, pages 1–8. Research Studies Press, 1994.