Workshop
Software on Demand: Issues for RE

Stephen Fickas
University of Oregon
fickas@cs.uoregon.edu

Introduction. Software on demand is a means of delivering software over the internet on an as-needed basis. The model is one where a user can download full apps or small plug-ins to complete the current task at hand. There are interesting features of this model:

- Throw away software. After it's used, software may be thrown away. The latest, most bug-free version can be retrieved again when needed. This opens up new payment schemes—software can be rented or even follow a pay-for-use approach.
- Niche marketing. It may be possible to develop software more tailored to individual needs.
- Software push/pull. The user may request new software (pull) or software vendors may attempt to sell the user on new versions (push).

A Strawman System. A prototype software on demand system, built by a team from the University of Oregon, will be used as a strawman for discussion. The prototype system was built with COTS in mind—an effort was made to use publically available software. The final system brought together the following pieces:

1. A base application. The example chosen was an information system for the University of Oregon campus. The base system, which could be given away free or at low cost, provided a map of campus and window panes for new components to work within. The expectation is that the base application would run on a user's home computer. The user would ask (demand) new components on an as-needed basis, e.g., if the user wanted information on computer labs on campus, she would ask for a specialized component to be added to the base app. The base application was written in Tcl/Tk for WIN95.
2. A component warehouse. Components were small pieces of functionality that could be added, on demand, to the base application. A general goal was that component addition would be without user intervention, e.g., no recompilation nor restart of the app would be necessary. The component warehouse was implemented as a database server on the Internet. Components were small Tcl/Tk programs.
3. A matchmaker service. This is the heart of the system. It is the piece that matches up a user's needs with the component that will meet those needs. Our approach was to use a modification of the reuse facets suggested by Prieto-Diaz. A user would select keywords from facet lists, and the matchmaker would index into the component warehouse for matches. The matchmaker was implemented as a web form and associated cgi scripts.
4. Online payment system. To close the loop, a means of paying component writers for their components was needed. The general goal was to transfer funds from user to component writer on download of a component. We chose to use First Virtual as the payment scheme.

We are able to pull all of these pieces together into a working system. We concluded that (1) the technology is largely in place to handle the "transport layer" (components, money) of a software on demand system, but (2) the question of matching user needs with available components is a wide open issue. Item (2) led to the organization of this workshop.

Some Workshop Questions. We list some questions that have been raised in our explorations of the software on demand idea.

1. How do end-consumers state their software requirements demands so that software can be found and brought to them? Such requirements will likely include a healthy dose of nonfunctional requirements having to do with price, guarantee and deliverability.
2. Does the software reuse community having anything to say in the software on demand area? We chose to use a reuse methodology for the prototype system. Should we continue to look to the reuse field for new results?
3. Can we eliminate humans from the RE process? Can software running on the user's machine detect the need for a new app or plug-in and write the requirements for it automatically?
4. How can software be organized on the net so that it can be found and downloaded on demand? How can vendors advertise the features

1090-705x/97 $5.00 © 1997 IEEE

(or even existence) of their software? Is there a role for negotiation between user and vendor, maybe through intermediate brokers?

We expect other questions will surface at the workshop itself. In summary, this is a new area with little prior work to base judgements on. We expect a lively and open discussion.