Web Programming with Continuations

William E. Byrd*

November 20, 2002

Contents

*Author's e-mail address: emacs_groks@yahoo.com

1 The Big Idea

Scheme supports first-class continuations, which can be resumed an arbitrary number of times. These continuations can be used to make HTTP behave like a stateful protocol, and can even eliminate the need for session time-outs.

Although continuation-based approaches to Web programming looks promising, several important issues related to the use of continuations still need to be addressed before these techniques will gain commercial acceptance.

2 The Problem

Since HTTP is a stateless protocol, Web applications themselves must maintain conversational state with each client. The usual approach is to break the application into a number of small pieces (JSP's, Java Servlets, CGI scripts, etc.), each of which is capable of handling a small number of requests.

This approach suffers from several problems:

- legacy applications not written in this style are difficult to adapt to the Web
- breaking the application into pieces may make it harder to modify the application
- screen flow is difficult to handle in general, as users can clone the browser window, submit simultaneous requests, bookmark pages, or hit the "Back" button on the browser
- techiques to maintain state often require each piece of the Web app to manipulate objects in a "session" hash table, resulting in side-effects and dependencies that can cause subtle bugs

3 Using Continuations to Add State to the Web

Since a continuation is a first-class value in Scheme and can be invoked an arbitrary number of times, continuations can be used to add state to a Web application. Whenever the Web application needs input from the user, the app saves the current continuation associated with that user. When the user responds with some information, the saved continuation is restored, and the input provided by the user is returned as the value of the continuation.

Since a continuation can be invoked multiple times, a user can bookmark a page and return to the page later. As long as the continuation still exists, the user can begin interacting with the application at that point in the "conversation".

For example, imagine someone who frequently uses an airline reservation site to fly the same route each week. The site might require that the route information be entered by the user each time a flight is booked. To save time, the user may wish to fill in the information once, then bookmark the next page of the transaction. As long as the continuation saved at that point is still valid, the user can go back to the site in the future and book a new flight without having to enter the route information again.

Note that in this case the user is able to save a pending transaction at an arbitrary point not explicitly supported by the designers of the Web site.

4 Issues to Consider

You will need to consider the following issues before using continuations in your Web application:

4.1 Saving the continuation

There are three basic approaches to saving continuations: save them on the server, on the client, or make the decision at run-time.

4.1.1 Save it on the Server

This approach reduces bandwidth, but has implications with respect to storage space and session timeouts. The saved continuation is usually mapped

to a unique URL by which it is invoked.

4.1.2 Save it on the Client

Once again, there are a couple of options here. The general tradeoff is that you reduce storage requirements on the server at the expense of increased bandwidth. In addition, you must be able to serialize/unserialize your continuation in some way.

1. Save the continuation in a cookie

   The advantage is that the continuation is persistent, eliminating session timeouts. A disadvantage is that you must deal with the submittal of older versions of continuations. Also, most browsers limit the size of cookies to a few kilobytes.

2. Save the continuation in a hidden field

   This may be the only option for saving larger continuations on the client-side. Session timeouts are still a problem.

4.1.3 Hybrid Approach

Look at the size of each continuation before deciding where to save the continuation. This approach combines some of the benefits of the previous two approaches, at the expense of additional complexity.

4.2 Simulating Continuations in Other Languages

Even if you are programming in a language that doesn't have explicit continuation support, there are still ways to use some of these techniques. See "Automatically Restructuring Programs for the Web" (Graunke, et al., 2002) for details on CPS, lambda lifting and defunctionalization.

Continuations can be simulated in Java by using exception handling, as is done in the Kawa Scheme interpreter. Unfortunately, these fake continuations are strictly less powerful than the real thing, and cannot be used to simulate co-routines, for example. For details, see

http://www.delorie.com/gnu/docs/kawa/kawa-tour_19.html

5 Advantages

Using continuations to add conversational state to a Web application has numerous advantages over traditional mechanisms:

- Can structure the program to match the problem.
- Gracefully handles unusual navigational patterns.
- Can use standard development tools, including debuggers.
- Can port legacy software to the Web more readily.
- Provides for a more rigorous understanding of program structure and Web navigation.
- Can store conversational state on the client's browser, if desired.
- Can take advantage of standard automatic program transformations.

6 Limitations

Unfortunately, all is not sunshine when using continuations in your Web app. Here are a few real disadvantages of a continuation-based approach:

- Continuations seem to be hard to understand.
- What if your language doesn't support first-class continuations?
- Where do you save the continuation?
- Garbage collection:
  - How long before you empty the trash?
  - How to handle distributed garbage collection?
- Must a continuation be associated with a unique thread?
- How do you reconcile continuations with persistent stores?
- How do you manage software versioning with continuations?

- If you use CPS, will anyone be able to understand your code?
- Will ease of implementation drive the user experience?
- Are continuations efficient?
- How do you convince your boss that this approach works?

7 Some Systems in Use

Here are a few Web-based systems that use either continuations or CPS to help maintain state:

Yahoo! Store http://store.yahoo.com/ (formerly *ViaWeb*)

See http://www.paulgraham.com/lib/paulgraham/bbnexcerpts.txt for details.

Persistent Server-Side Scheme Interpreter (PS^{3}I)

(http://youpou.lip6.fr/queinnec/VideoC/ps3i.html)

See the papers by Christian Queinnec in the *Further Reading* section of this paper for more information on PS^{3}I and the Université Paris CD-ROM.

Université Paris CD-ROM (http://videoc.lip6.fr/)

HTDP Web Server (http://www.htdp.org)

See the papers by Paul Graunke, et al., for details.

Other Systems See the comp.lang.scheme thread I started on November 19, 2002.

8 Open Problems

As I see it, the three biggest technical hurdles preventing adoption of continuation-based Web programming in the commercial sector are:

1. dealing with side effects, especially mutable stores

2. determining the best manner in which to save continuations

3. designing systems so that the GUI does not receive sloppy seconds

Graham may have solved some or all of these problems in designing ViaWeb - I'll ask him.

9 A Modest Proposal

If the Metro-Schemers group decides to create a Web site, I propose that we use a continuation-based architecture. This would better allow us to see how a continuation-based system works in practice.

One possibility might be to use the DrScheme TeachPack mentioned in "Automatically Restructuring Programs for the Web" (Graunke, et al., 2002).

10 Further Reading

Perhaps the best source for in-depth information about Scheme is Jim Bender's "Bibliography of Scheme-related Research" at http://library.readscheme.org/

Most relevant to this presentation is Bender's "Reading list on XML and Web Programming" at http://readscheme.org/xml-web/ and his "Continuations and Continuation Passing Style" page at http://library.readscheme.org/page6.html.

Unless indicated otherwise, links to all of these papers are available on Jim's site.

10.1 Continuations

The following books and papers contain much useful information on continuations:

- R. Kent Dybvig. *The Scheme Programming Language* (2nd ed.). Prentice Hall, 1996.

  Entire text available online at http://www.scheme.com/tspl2d/index.html.

  Sections 3.3 and 3.4 introduce continuations and continuation-passing style, respectively.

- Paul Graham. *On Lisp.* Prentice Hall, 1993.

  Entire text available online at http://www.paulgraham.com/onlisptext.html.

  Chapter 20 contains an in-depth examination of Scheme continuations.

- Robert Hieb, R. Kent Dybvig, and Carl Bruggeman. "Representing Control in the Presence of First-Class Continuations". *ACM SIGPLAN 1990 Conference on Programming Language Design and Implementation.* June 1990.

  This paper examines how to efficiently implement continuations, which may provide the reader with additional insight into the nature of continuations.

- John Small. "Introducing PLT Scheme." Pea Pod Publishing (Rogare L.L.C.), Great Falls, Virginia, 2002.

  Available at http://www.rogare.com/metro-schemers/intropltscheme.pdf.

  Chapter 3 contains an in-depth look at continuations.

10.2 Web Programming with Continuations

These documents show how continuations and continuation-passing style can be useful in Web application development:

- Paul Graham. "Lisp in Web-Based Applications". Excerpt of talk given at BBN Labs in Cambridge, MA, April 2001.

  Available at http://www.paulgraham.com/lib/paulgraham/bbnexcerpts.txt

  Graham briefly explains how the architecture of ViaWeb exploited continuation-passing style.

- Paul Graunke, Robert Findler, Shriram Krishnamurthi, Matthias Felleisen. "Automatically Restructuring Programs for the Web". *Automated Software Engineering 2001.* 2001.

  This paper explains how to use several automatic program transformations (including CPS) to allow Web-based programs to be written in a simpler fashion than usual.

- Paul Graunke, Shriram Krishnamurthi, Van der Hoeven and Matthias Felleisen. "Programming the Web with High-Level Programming Languages". *Proceedings of ESOP 2001.* 2001.

  Explains how features of a high-level programming language, including first-class continuations, allow for simple yet efficient creation of Web applications.

- Christian Queinnec. "Inverting back the inversion of control or, continuations versus page-centric programming". Technical Report7, LIP6, May 2001.

  Avaliable at http://youpou.lip6.fr/queinnec/PDF/www.pdf.

  One of several Queinnec papers on continuations and the Web. This paper contains some interesting implementation details.

- Christian Queinnec. "The influence of browsers on evaluators or, continuations to program web servers". *ICFP '2000 - International Conference on Functional Programming* pp. 23-33, Montreal (Canada), September 2000.

  Available at http://youpou.lip6.fr/queinnec/PDF/webcont.pdf.

  This paper explains how to save continuations on either the client-side or the server-side to maintain conversational state in a Web-based system.

- Christian Queinnec. "A library for quizzes". *Workshop on Scheme and Functional Programming (2002).* October 2002.

  This work follows up on Queinnec's ICFP 2000 paper, but does not focus on continuations per se.