

In the Web of Generated “Clones” (Position Paper)

Holger M. Kienle and Hausi A. Müller
Computer Science Department
University of Victoria
Victoria, Canada
{kienle,hausi}@cs.uvic.ca

Anke Weber
ExperEdge
Technology Partners
Victoria, Canada
weber@experedge.com

Abstract

In order to be of use to a software maintainer, the clones detected by a tool should provide meaningful information. However, generated code can diminish the usefulness of clone detection significantly because generated “clones” are reported that are of no interest to the maintainer. We discuss this problem for the Web domain, using a Web site that we developed as an example to further illustrate the problem.

1. Motivation

Clone detection approaches typically assume that clones are introduced manually by programmers that first copy and subsequently modify a piece of existing code. These clones are viewed as undesirable because of their negative impact on software maintenance. Clone detection techniques have been developed for procedural as well as object-oriented software. More recently, this work has been expanded to include Web sites. Web sites can contain clones in HTML, scripts such as JavaScript and Perl, XSLT, XML, DTD, XSchema, etc.

Automatic clone detection can be a useful aid for software maintenance and reverse engineering. For example, the detection of clusters of similarly structured HTML pages is a good starting point to refactor a Web site to a design that separates content and navigational structure. However, to be effective, the detected clones have to be sufficiently precise and meaningful in the eyes of the person inspecting the proposed clones.

In this context, a particular problem arises for software that is partly hand-written and partly generated automatically. Software systems that consist of generated components tend to generate highly regular—sometimes even identical—pieces of code. Automatic clone detection sys-

tems will flag them as clones, but they are, in fact, spurious clones that are of no interest to the software maintainer who understands the build process of the software.

Generation of code is accomplished by approaches such as domain-specific languages, model-driven architecture (MDA), and dynamic Web technologies. Generative techniques are far from being a new idea (yacc was developed in the early 1970’s), but they are currently receiving renewed interest. The OOPSLA 2003 conference has for the first time a special track on “Domain Driven Development,” which discusses various generative techniques. Tu and Godfrey discuss the generative aspects of GCC and Perl [4]. Generative techniques are also found in UML tools (e.g., ArgoUML), GUI builders (e.g., Borland Delphi), and Web development environments (e.g., Dreamweaver and WebSphere Application Developer).

In this paper we focus on generative techniques in the Web domain—one can imagine similar issues for traditional software systems. We briefly describe a Web site that we created (BSENG) and how the generative techniques that we used pose challenges for clone detection tools.

2. A Web Scenario: The BSENG site

The BSENG Web site under discussion has been developed to promote the new Bachelor of Software Engineering at the University of Victoria [2]. We decided to build BSENG with Macromedia Dreamweaver since it is a mature, commercial tool.

In order to simplify maintenance and to ensure a consistent layout, we used Dreamweaver’s *template* mechanism. If a page is based on a template, the page designer is allowed to edit only certain parts of the page. Later on, Dreamweaver automatically generates the code for a page by using the predefined template and filling in page-specific information into template slots. The BSENG site uses one template to defined look-and-feel and another template for

the print view, which renders the information of each page without graphical embellishments and navigation panel.

The BSENG site makes also use of Dreamweaver's scripting, which exposes tool functionalities with a programmable API. These *API scripts* can be used to automate recurring development activities. The pages for BSENG's print view are generated off-line with an API script that is invoked by the developer before deploying the site.

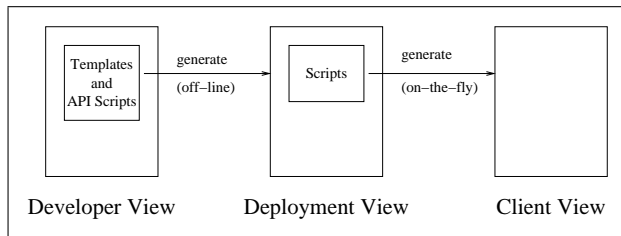


Figure 1. Web application views

For Web applications, we can distinguish between three different views (see Figure 1):

client view: The view of the Web site that a client (typically using a Web browser) sees.

deployment view: The view of the Web site that a Web server (accessing the local file system) sees.

developer view: The view of the Web site that a developer (using a development tool such as Dreamweaver) sees.

Often, generative techniques make transformations from one view to the other. Both Dreamweaver's templates and API scripts are examples of transformations that happen during the transition from the developer to the deployment view. Dynamic server-side scripts (e.g., JSP or servlets) perform transformations between the deployment and the client view. (The BSENG site currently is static and does not use such scripting.)

Different tools and technologies operate on different views. It is important for the developer to understand the differences between these views in order to interpret the information of the views correctly. Clone detection techniques can operate on each of these views. The tools that we are aware of operate on the deployment view [1, 3]. For the client view, a clone detection tool would have to crawl the Web site similar to a Web spider. Finally, one can also envision clone detection tools as extensions to existing Web development tools. Several such tools (e.g., Dreamweaver and GoLive) can be extended and customized via scripting. This approach requires an implementation that is customized to a particular development tool. Here, the clone detection algorithm operates on the same view as the developer and hence can account for special (generative) concepts such as templates.

In the BSENG site, the use of templates and API scripts pose problems for clone detection. In general, if similar Web pages are generated from a common template, a clone detection analysis (operating on the deployment or client view) will potentially report a large amount of cloned material. These "clones", however, are of no concern to the maintainer who works at a higher level of abstraction in the developer view, which exposes the templates. Templates are supported, for example, by GoLive, Dreamweaver, and Wikis.¹ Note that templates can cause "clones" in HTML, server-side scripting, and client-side scripting.

A similar problem arises with our API script that constructs the print view for each page. The script effectively copies each HTML file to a new director `pages4print`. A new template is associated with these pages; otherwise, the original and the copied page are identical. In the deployment (and developer view), the pages differ only by a link that points to a different style sheet. Thus, it is quite likely that they will be classified as clones.

3. Conclusions

For a Web site, we can identify three important views in the build process (cf. Figure 1) that a Web developer and maintainer has to work with. We believe that clone detection techniques could benefit from an understanding of these different views. Clone detection tools could then focus on clones relevant to the particular view and automatically suppress spurious clones caused by generated code.

References

- [1] G. A. Di Lucca, M. Di Penta, A. R. Fasilio, and P. Granato. Clone analysis in the web era: An approach to identify cloned web pages. *Seventh IEEE Workshop on Empirical Studies of Software Maintenance (WESS 2001)*, pages 107–113, Nov. 2001.
- [2] H. M. Kienle, A. Weber, J. Martin, and H. A. Müller. Development and maintenance of a web site for a bachelor program. *5th International Workshop on Web Site Evolution (WSE 2003)*, Sept. 2003.
- [3] F. Lanubile and T. Mallardo. Finding function clones in web applications. *Seventh Conference on Software Maintenance and Reengineering (CSMR 2003)*, pages 379–386, Mar. 2003.
- [4] Q. Tu and M. W. Godfrey. The build-time software architecture view. *International Conference on Software Maintenance (ICSM 2001)*, pages 398–407, Nov. 2001.

¹Interestingly, Dreamweaver can export HTML comments that function as meta-data to identify templates in the generated HTML page. A dedicated clone detection algorithm could make use of this information to suppress clones caused by templates. However, the exporting of meta-data information is typically disabled to optimize bandwidth in the deployed version of the Web site.