Software Reuse at Hewlett-Packard

Martin L. Griss
Software and Systems Laboratory
HPL-91-38
March, 1991

software reuse, software process, software construction, corporate reuse program, hypertext, libraries, application frameworks

In this position paper, several software reuse related activities at Hewlett-Packard are described, focusing on those in which the author has been involved over the past 6 years. Recent activities include an investigation (started in November 1989) of the opportunities for a systematic, HP Corporate Software Reuse program that led to a new HP Corporate Engineering project to define and execute such a program. Planned work involves a combination of consulting, training, methods development, the writing of a reuse handbook, and several pilot projects. Also related are two research projects at HP Laboratories, one involving the development of a prototype hypertext-based reuse library management system (Kiosk), and the other (FAB) exploring component-based software construction using application development frameworks for distributed applications.

1 Introduction

It is widely believed that a systematic application of reuse to prototyping, development, maintenance and software process is one of the most effective ways to significantly improve the software development process[BE89, Tra88]. Software reuse simultaneously offers improved time-to-market, software quality and application consistency, and reduced development and maintenance costs[BB91].

New work in software reuse such as domain analysis[PD90], object-oriented methods, library technology[FG90] and architectural frameworks promises to lead to a consistent methodology that will be useful for domain specific reuse. It is also important to jointly consider software process and software reuse[CHSW90], and how open-architectured software engineering environments are built and tuned to support domain-specific reuse processes[Boe90].

However, a cursory investigation of the status of reuse in many companies, such as Hewlett-Packard (HP), indicates that while several groups are making progress with some aspect of reuse, very few software practitioners see reuse as a major or systematic part of their software process. Recently, several companies have started to report significant success with their reuse programs. HP also is involved in understanding and applying the state of the art and practice to remain competitive.

I have been directing software technology and software reuse projects at Hewlett-Packard Laboratories (HP Labs) for over 6 years, with a recent focus on reuse-oriented research in HP Labs. Since November 1989, as consultant to HP Corporate Engineering's Software Initiative, I have studied the status of software reuse in HP, and exploring opportunities to make software reuse a significant part of HP's software process for the 1990's. We believe that a fairly broad, well coordinated software reuse program involving management, process and technology is needed to make significant progress.

Our reuse-oriented work in Corporate Engineering and HP Labs is directed at two major goals:

- Making systematic software reuse a significant part of HP software development processes.
- Research in the area of reuse library systems and reuse-based applications development.

My activities supporting these goals involve:

- Consulting with HP Corporate Engineering on software reuse and several other software process issues, to help define a coherent set of software initiatives aimed at improving HP software productivity and quality, and at reducing time to market and maintenance costs.
- Investigating reuse inside and outside HP via an extensive e-mail survey, attending several workshops and tutorials, and investigating reuse activities in several companies and universities. The results were presented in talks and tutorials to HP upper management and engineers during 1990.

- Defining and (since October '90) helping execute a Corporate Software Reuse program, which will involve the investigation of impediments to effective reuse, the development and acquisition of reuse methodology technology, the ongoing assessment of HP's progress towards systematic reuse, and the discovery and communication of best practices.
- Leading a small HP laboratories project team to develop a hypertext-based reuse library prototype for software libraries. In October 1990 we released the first prototype to internal HP users.
- Leading a small HP laboratories project doing research on methods of rapid application development based on reuse and user programming, building upon application- and reuse-oriented architectures and frameworks.

More details on several of these activities are given in the following sections.

2 HP's Experience with Software Reuse

Hewlett-Packard has an extensive history with software reuse, and currently has several active projects in its divisions and laboratories. The following gives the flavor and scope of HP's projects. Several have developed or use existing DBMS systems to store and distribute software components.

- Early work involved the development and networked distribution of a family of instrument modules (in Instrument Basic).
- Subsequent work used Objective-C to develop class libraries, tools and metrics. Human-Interface classes and common instrument sub-systems have been widely distributed within the company, and some have been provided outside. HP has also developed and distributes Motif 'widget' libraries.
- More recent work involves several multi-division domain analyses and common architecture and library projects for several HP groups, such as embedded software for instruments, and systems architecture for chemical and medical systems. Some of the work involves generic application frameworks and major components to be used in several products.[1]
- Several other divisions are involved with various aspects of the reuse process, setting up libraries, establishing reuse goals (such as deposition ratios, or reuse levels), and establishing and running reuse councils or component projects.
- Several divisions have been developing C++ extensions, tools and class libraries for instrument systems and general-purpose use.
- At HP laboratories, we are researching object-oriented analysis and design, reuse tools, domain-oriented frameworks, user-programmable systems and components, and various kinds of object-oriented systems and tools.

[1]Our Corporate Reuse program has begun to consult on domain-analysis, architecture and object-oriented methods to some of these projects

3 Corporate Reuse Program

We have designed a Corporate Reuse program that will consist of a core team of software reuse experts in Corporate Engineering, with several additional people working on assignment with a selection of pilot projects in the divisions. We do not plan to build a single Corporate reuse library (though we might provide a single library mechanism, perhaps with a few key components "pre-loaded"). This focus seems most appropriate to HP's culture and state of reuse and process maturity.

The core team will be involved in:

- The development and execution of a reuse assessment (questionnaire and metrics), to determine how different groups are progressing, and to diagnose various people, management, process, technology and readiness issues.
- The design and execution of several "pilot projects" with divisional partners to develop and test methods and tools for Domain Analysis, Application Architectures, Design for Reuse, Development with Reuse, and Library Management.
- The collection and dissemination of reuse best practices, processes and guidelines in a handbook, customizable to the process and needs of different divisional reuse efforts. We will test and refine these guidelines in the context of the pilot projects.
- Consulting as a team of "resident experts" to the pilot and other reuse projects in HP.
- Helping develop and coordinate training and other technology transfer activities with the assistance of a Corporate Engineering technology transfer unit, and other management and engineer software training, workshop and publication programs.
- Coupling and coordinating the reuse program to other Corporate Engineering Software Initiative programs (maintenance, configuration management, prototyping, process and metrics).

This program is just getting underway: we are staffing the core team, we have started assessing several divisional reuse programs, and we held a first Reuse Practitioner's Workshop for 60 HP engineers and managers in February 1991. We expect to select the first of several pilot projects in April, and will refine our plans as we proceed.

4 Hypertext-based Software Reuse Tool

Much of our work on reuse-based toolsets at HP Labs is guided by a view that hypermedia-based systems are the most appropriate frameworks to integrate tools, maintain links between all software workproducts (such as specifications, requirements, source code, header files, documentation, test files, build scripts, etc..)[Big89, FHR91, Car90, LJ88]. Appropriately combined with a configuration management system, hypertext also offers a structure to manage the evolution of a developing system and its supporting component libraries[LFB89, GS90]. Thus we are interested in integrated tools for both reuse and total software lifecycle support.

Several tools for C++ programming, particularly browsers and graphical aids, were originally developed in our laboratory, and further developed by partner divisions. Of some interest is a tool (now available as an HP product, SoftBench C++ Developer[AB90]) that keeps several levels of abstraction compatible, allowing a library of C++ code, header files and partially completed software to be managed consistently.

The Reuse Tools project team in HP labs is developing a workstation based reuse toolset, using C++ and InterViews[LVC89] as a base. The initial goal of this prototype (called *Kiosk*[CFG91]) is to provide a hypertext framework for manipulating (object-oriented) libraries, such as InterViews, or other C and C++ libraries. The prototype is well integrated with the Unix file system and Unix tools, providing us with a flexible base for experimenting with several kinds of reuse-oriented software development and software management environments.

Kiosk includes an import tool (called Cost++) which uses a simple declarative language to describe and mechanically build links and nodes to help classify, catalog and structure the relationships between several software "workproducts" (software lifecycle artifacts). We have used Cost++ to build several different kinds of reuse library structures, on top of InterViews and an internal HP library. We have also used Cost++ to develop a hypertext interface to the Kiosk documentation, producing a rudimentary on-line help system.

These classification and catalog nodes, and the original workproducts are then accessed from a browser/editor, which provides both graphical and textual access and editing/annotation capabilities. We have also developed a simple query language, which is integrated with the browsing mechanism to produce a filtering/finding environment - queries produce hypertexts which can then be further browsed, combined, pruned, or otherwise manipulated.

We have also explored some simple IR-based mechanisms to extract keywords, to automatically create links between interesting places in several workproducts, and to experiment with simple automatic classification. This has also been applied to the on-line Kiosk help/documentation browser.

User and visitor comments on the role of hypertext for component library management and our initial Kiosk implementation are favorable. We have received several interesting suggestions about increasing the scope of Kiosk application, such as using it to help in managing the terminology and artifacts produced during domain analysis, or combining Kiosk with a faceted approach.

Kiosk 1.0 was first released to HP internal users in October 1990. We have made several improvements to the system based on user feedback, and released an updated version in January 1991. At this point some 40 internal users have installed Kiosk. Several of the users are exploring how Kiosk might manage workproducts other than code, such as network management interfaces, or design fragment and decisions.

Nevertheless, we have yet to show that a hypertext-based library does, in fact, help make reuse easier. Over the next several months, we plan to evaluate the prototype in use by groups doing library-based reuse.

5 Framework-based Application Construction

Another project team (called *FAB*) is exploring a "kit" building technology to produce a series of compatible, application-oriented kits. These kits (reuse library, glue language, application framework and supporting environment) will be used to produce distributed applications that are easy to customize and extend, and which can be usefully and easily integrated together (by an application developer or end user) to produce application-oriented environments. As a first step, we have begun to explore the structure and use of "software bus" architectures[Rya90], distributed applications based on a broadcast message server (SoftBench BMS[Cag90]), and several object-oriented application frameworks.

A key feature is a common application architecture and compatible or common extension language(s), with support tools that can be easily and effectively used by application developers and end-users. We believe that effective kits need both an execution environment supporting application use and customization, and a development environment that is oriented to the reuse/glue paradigm (with visual builders, library browsers, component interconnectors, etc.).

We believe that these execution and development environments can (and must) share some common technology, mechanisms and philosophy. For example, systems such as Hypercard, AutoCAD and others actually combine both environments, and use the same scripting language (visual and textual) as both extension language and application development language. This allows a spectrum of usage, ranging from "simple end user" through "experienced customizer" to "advanced application package writer".

Issues explored in these projects include:

- Design - domain analysis and object-oriented methods; architectural reference models; generic framework and architectures; distribution mechanisms; broadcast message and software bus models.
- Reuse - component design and framework API's; library structure and taxonomy(s); component interconnection mechanisms; parameter transmission; extension mechanisms; generic "kit" technology.
- User Programming - compatibility of textual and visual paradigms; "glue" languages; generic customization languages; metaphors for how library and extension structure reveal themselves (e.g. cut-and-paste from palette, versus declarative 4GL code, etc).
- Environments - library browsers and code-writers; hypertext linkages between different workproducts; debugging support for glue/scripting languages; version management.

6 Summary

We have several reuse related projects underway at Hewlett-Packard. We believe it important to make progress on several areas simultaneously. In order to make systematic reuse a significant part of the way we develop and maintain software at HP, we are defining a comprehensive and coherently managed program, involving methodology and technology development and transfer, assessment and training, and planned change management. We have good management support for our program and are making significant progress.

7 References

[AB90] Michael Armistead and John Burnham. HP C++/Softbench: A development environment for C++. *Journal of Object-Oriented Programming*, 3(4):82–85, November 1990.

[BB91] Bruce Barnes and Terry B. Bollinger. Making reuse cost-effective. *IEEE Software*, 8(1):13–24, January 1991.

[BE89] Ted Biggerstaff and Alan Perlis (Eds). *Software Reusability*, volume 1 & 2. ACM Press, NY, 1989.

[Big89] Ted J. Biggerstaff. Design recovery for maintenance and reuse. *IEEE Computer*, pages 36–49, July 1989.

[Boe90] Barry Boehm. Trends in US environments. In Richard N. Taylor, editor, *Proceedings of Fourth ACM SIGSOFT Symposium on Software Development Environments, Dec 3-5, 1990, Irvine, CA*. ACM, ACM Press, December 1990. (Keynote address, not printed in proceedings.).

[Cag90] Martin R. Cagan. The HP Softbench environment: An architecture for a new generation of software tools. *Hewlett-Packard Journal*, pages 36–47, June 1990. (See other papers in this issue.).

[Car90] Patricia Carando. Hyperbole: A retrieval-by-reformulation interface that promotes software visibility. In Doug Lea, editor, *Third Annual Workshop: Methods and Tools for Reuse*. CASE Center, Syracuse University, June 1990.

[CFG91] Michael L. Creech, Dennis F. Freeze, and Martin L. Griss. Kiosk: A hypertext-based software reuse tool. Technical Report SSL-TM-91-03, Hewlett-Packard Laboratories, March 1991. (Preliminary version).

[CHSW90] Joachim Cramer, Heike Hanekens, Wilhelm Schafer, and Stefan Wolf. A process-oriented approach to the reuse of software components. Memo Nr. 43, University of Dortmund, Helenenbergweg 19, D-4600 Dortmund 50, FRG, March 1990.

[FG90] William B. Frakes and P.B. Gandel. Representing reusable software. *Information and Software Technology*, 32(10):653–664, December 1990.

[FHR91] Gerhard Fischer, Scott Henninger, and David Redmiles. Cognitive tools for locating and comprehending software objects for reuse. Technical report, University of Colorado, Boulder, May 1991.

[GS90] Pankaj K. Garg and W. Scacchi. A Hypertext System to Manage Software Life Cycle Documents. *IEEE Software*, pages 90–98, May 1990.

[LFB89] E. Lippe, G. H. Florijn, and E. G. J. Bogaart. Architecture of a distributed version control system. Technical Report RP/dvm-89/4, SERC, P.O. Box 424, 3500 AK Ultrecht, The Netherlands, April 1989.

[LJ88] L. Latour and E. Johnson. SEER: A graphical retrieval system for reusable Ada software modules. *Third International IEEE Conference on Ada Applications and Environments (Cat. No.87CH2470-3)*, pages 105–113, May 1988.

[LVC89] Mark A. Linton, John M. Vlissides, and Paul R. Calder. Composing user interfaces with InterViews. *IEEE Computer*, pages 8–22, February 1989.

[PD90] Ruben Prieto-Diaz. Domain analysis: An introduction. *Software Engineering Notes*, 15(2):47–54, April 1990.

[Rya90] Doris Ryan. *RAPID/NM, Reusable Architectures for Transaction Processing and Network Management Applications.* AT&T, 1990.

[Tra88] Will Tracz. *Tutorial: Software Reuse: Emerging Technology.* Number IEEE Catalog Number EH0278-2. IEEE Computer Society Press, 1988.