Corporate Source: Applying Open Source Concepts to a Corporate Environment (Position Paper)

Jamie Dinkelacker, Pankaj K. Garg
Software Technology Laboratory
HP Laboratories Palo Alto
HPL-2001-135
May 31st , 2001*

E-mail: garg@hpl.hp.com

Open Source, software engineering, software reuse, software process, library

Corporate Source harnesses the power of the Open Source (TM) development method for HP. The basic idea is to make available source code from various HPL projects to members of the HP software engineering community. This is analogous to the research library's current processes of making available HPL technical reports within the company.

Using the Corporate Service, members of HPL can "publish" their software such that it can be browsed and used by anyone inside HP. Over time, in an Open Source (TM) manner, submitted software will find use in other parts of the company and be improved by not just the original authors, but also by members of the HP software engineering community at large.

* Internal Accession Date Only Approved for External Publication
Presented at the 1st ICSE International Workshop on Open Source Software Engineering, May 15, 2001, Toronto, Canada

1 Introduction

Corporate Source is the application of Open Source concepts, perspectives, and methodologies within the corporate environment – i.e., "open" to all developers behind the firewall. The same benefits of developing according an Open Source model are then available to internal projects, although the participating community size is smaller than the Internet. Nevertheless, for a world wide company open Corporate Source brings a unique set of advantages to enterprise developers, albeit not without challenges of its own as described below.

Open Source software development process has proved itself with several industrial strength software products: e.g., the Linux operating system, the Emacs text editor, and the Apache web server. In each of these cases, Open Source has resulted in substantial benefits for software maintenance, reusability, and quality [3]. Often, these Open Source products are preferred by the market than their "Closed Source" counterparts. While the Open Source development method has various reasons for its success, one of the main ingredients was best summed up by Eric Raymond: "given enough eyeballs, all bugs are shallow."

From a software engineering perspective, this can be viewed as a natural progression of the code inspections process [1]. Unlike code inspection, however, the Open Source process is much more liberal and pervasive in that even the software user community gets involved in critiquing and reviewing source code.

While attractive from a purely quality perspective, the Open Source mechanism creates a fundamental conflict for software corporations: business practices dictate that software corporations retain Intellectual Property (IP) rights in their software, hide such IP from their competitors, and make profit on their investment in creating such IP. Realizing this business conflict, several Open Source derivatives have developed over the past few years. For example, Sun Microsystem uses a software Community Source license that is different from the Open Source license. Another approach is for corporations to expose some part of their products as Open Source while keeping the critical ones as proprietary source. An extreme form of this, which is the norm in the software industry, is to reveal only the user interface or *application programming interface* (API's) for software. When software corporations go for such source-code development processes, they lose out on the potential benefits of a possible Open Source counterpart. These benefits have the potential to be broad-ranging and cover the gamut not only of making bugs shallow , but speeding up time-to-market, improved software reuse, and rapid redeployment of skilled developers.

2 Corporate Source

For corporations, these conflicting requirements of Open Source and Closed Source can be addressed by a middle ground using a *Corporate Source* model of software development. This model requires a novel approach for large corporations to adopt for their software development: Instead of relying on a single-product, project-focused development method, this advance calls for a *corporation-focused* development method. With this method, each employee of the corporation can potentially contribute to the development of any given software product. By restricting the openness of the software development to within the corporation, the corporation does not incur the aforementioned

business costs of Open Source, nor incur any liabilities for having released untested software and the like. If the corporation is large enough (with a few hundred employees), it can realize the main benefit of Open Source, i.e., "given enough eyeballs, all bugs are shallow."

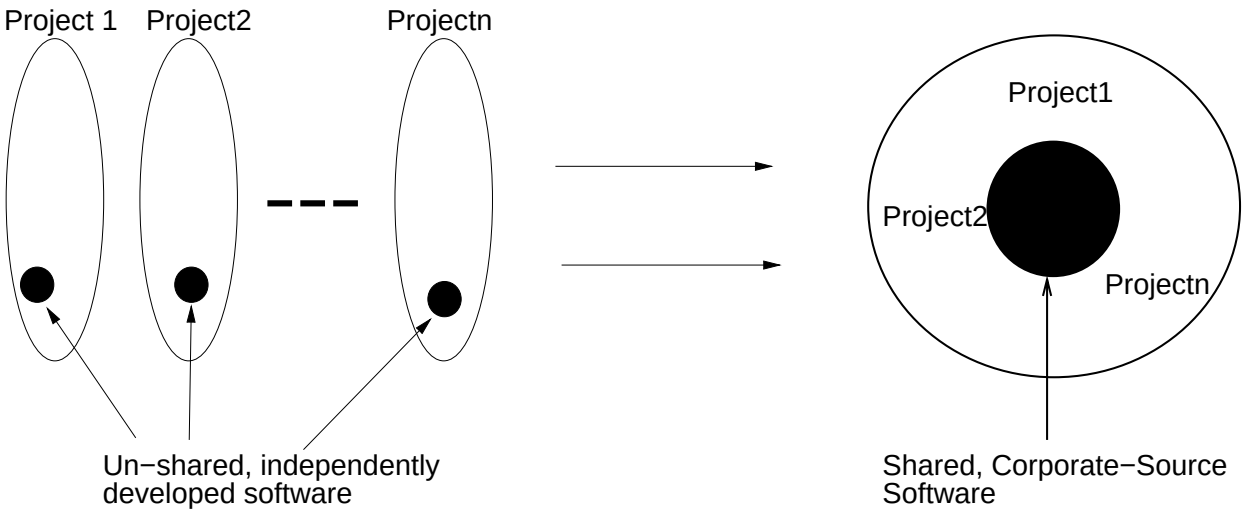

Figure 1: Transforming corporate Software Engineering

Corporate Source enables the following benefits for an organization:

- A readily available potpourri of software that can be built upon and used as starting point;
- Improved quality levels of shared software as authors' reputations are at stake;
- Shared, community debugging;
- Ability to easily integrate the corporate software development efforts into the overall Open Source movement, leverage the Open Source tools and methods, and ensure appropriate cross-learnings;
- Rapid redeployment of key developers from one project to another who already are familiar with the current Corporate Source code tree, tools, and coding standards; and,
- Faster development schedules with code leveraged among several products.

3 Challenges

Implementing corporate source within corporations has been – and will continue to be – challenging. Some of these challenges are organizational: e.g., how do we develop code across project and

organizational boundaries, and how do we identify and retain module designers. Other challenges are related to infrastructure technology. Both these aspects must be addressed in a satisfactory manner to ensure a successful and continuing deployment of Corporate Source.

Organizational challenges of Corporate Source include:

- Virtual Organizations: Most corporations today operate on a hierarchical organizational structure. When things work well, this can complicate the process of code sharing by having differing product roadmaps and timelines, where some managers may just push to get something delivered by a promised date, irrespective of code quality, which might then be an embarrassment to post into the Corporate Source code tree. When things don't work well, it's possible that some managers or even developers may be inimical to contributing any resources to perceived resource competitors within the organization.

- Leadership: The Open Source model depends on at least one *leader* or *owner* of given software modules. Such leaders are efficient designers and implementors for the kind of software module they are leading. In the open market of the Internet, *invisible hand* sort of mechanisms (based on visibility, ego, etc.) ensure that a good leader emerges for a given software module. In the case of Corporate Source, two distinct challenges arise: (1) what happens when a leader of a software module decides to leave the company, and several projects are critically depending on that module for their projects; or, (2) worse yet, no particular leader emerges for any given module?

- Task Assignments: The Open Source model depends on a willing and able force of capable software engineers who work on any given software module. The pool of people to draw from is the entire world population! (As programming is getting easier, the world of programmers is increasing.) For Corporate Source, the pool of programmers to draw from is whatever is available inside the corporation. Traditionally, project managers can determine their personnel skill requirements based on the requirements of their projects. In the Corporate Source model, however, the entire company's pool of programmers can potentially help out in the development of a given project's source code. How does one manage the appropriate skill set at the corporate level?

- Developer Indoctrination: The fundamental aspect of software development is the skill set of each individual developer. For Corporate Source, this necessitates that they be aware of the Corporate Source tree and tools, that they adopt the coding standards set forth to be consistent with the source tree, and that they develop a good judgment as to what constitutes a reasonable contribution to the source tree. These are organizational and managerial challenges to broad adoption and continued usage of Corporate Source within the enterprise. While many corporations do in fact have established and clear coding standards, this model can break down across divisions that serve distinctly different markets. This creates the challenge of how to maintain coding standards, how are new developers trained into using the Corporate Source.

The technology infrastructure requirements for Corporate Source include:

- Repository: An appropriate corporate-wide repository must be set up that can host the development of a large number of users. Adding (and updating) software in this repository must be easy and straightforward.
- Community Support: Open Source, and by hypothesis Corporate Source, thrives on active "fire-side" communities of developers [2]. Appropriate tools must be provided for such communities to develop around software projects in Corporate Source.
- Security: For aforementioned IP protection reasons, software developers and managers in large corporation will feel insecure about putting software in the "open," even if its within the confines of the firewall. The insecurity varies with projects from very loose security requirements to highly-sensitive company confidential information. A security classification and control system, much as the corporate technical reports system must be developed. The challenge is to assure the IP owners that indeed the security is enforceable.
- Search and Navigation: As with any other repository of such nature, efficient search and navigation systems must be developed.
- IS/IT Support: Corporate Source, like any server-centric content, requires hosting and maintenance of the code tree, platform, tool version control and related software engineering tasks. While often overlooked, the IS/IT support is absolutely crucial for maintaining uptime, running scheduled backups and recovery when necessary, and hardware maintenance as well.

4 Example: HP's Corporate Source

As a grassroots effort, Corporate Source began in HP Labs as a concerted effort to bring the benefits of Open Source into the internal software development community. It's worth noting that HP has many disparate software developers spanning the globe.

Figure 2 shows the main web page for accessing the Corporate Source service.

HP Corporate Source is currently hosted through the HP Labs Research Library. The library pages are familiar to HP software engineers and thereby combines together in a known place where many to seek information and tools. Appropriate IT functions provide for hosting, system maintenance, and backup.

The Corporate Source website is searchable, and shows quick summaries, with active links, of the most frequent downloads, and the most frequently viewed modules. It provides an easy mechanism to upload modules. The HELP feature shows briefly how to use the site, whom to contact, and displays publishing guidelines (i.e., coding standards). Each submitted software has a unique, one word software ID. Each software item lists one contact person, who is also the publisher of the software, although a software item can have multiple authors. The source code is stored in a configuration management system, called Concurrent Version System (CVS). The information about source code is stored in XML files. Both these reside on a Linux Corporate Source server. Also, a mail-reflector is used to keep developers apprised of Corporate Source topics and submissions.

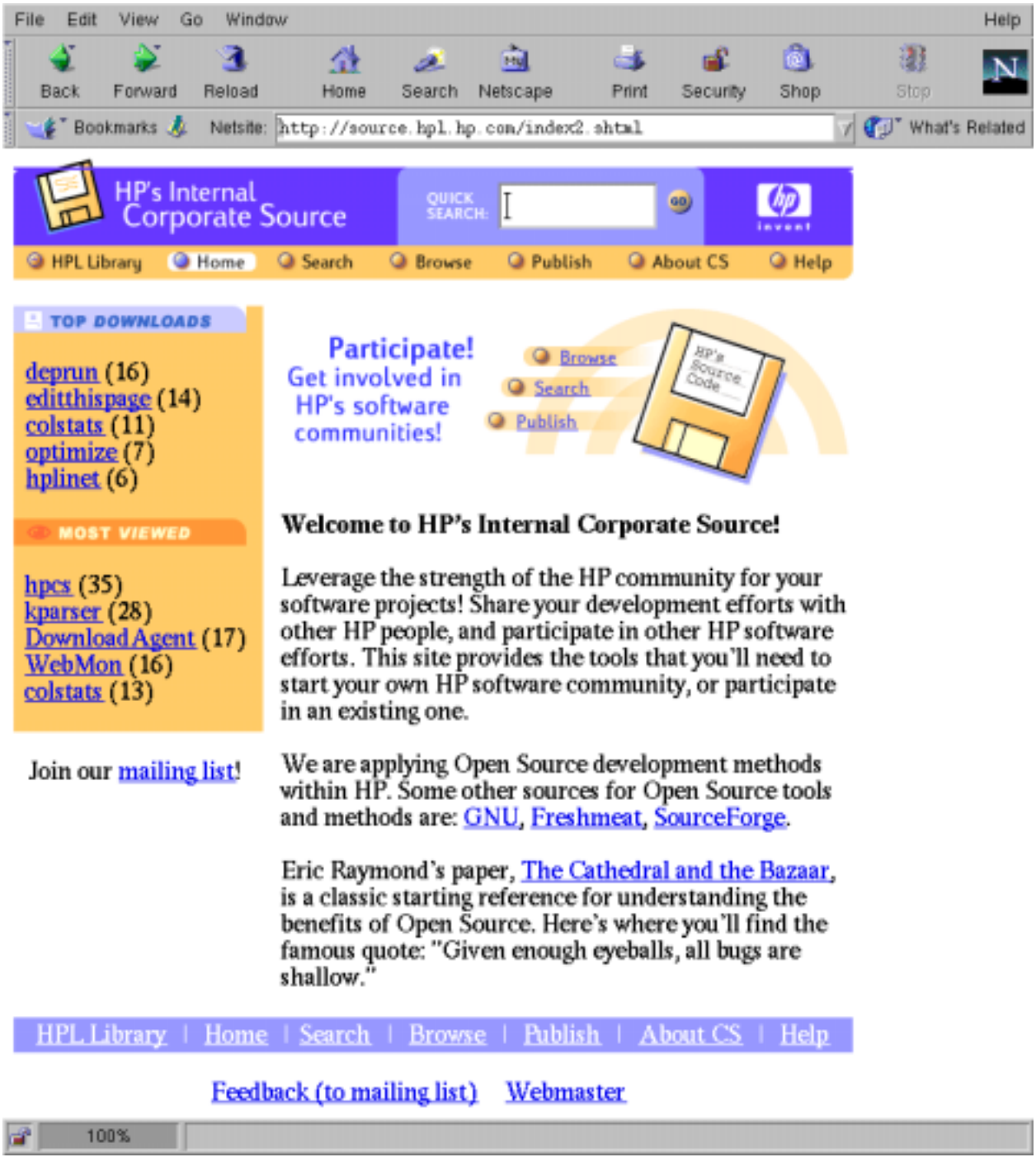

Figure 2: Main page for Corporate Source

It's worth noting not all HP developers are focused on generating software products – the company uses advanced decision technologies and supply chain management software as well. Much of this software is internally developed and won't necessarily be released into a product for obvious reasons related to competitive corporate performance. Yet, Corporate Source is well-suited for long-term evolution of these tools and capabilities. In other words, Corporate Source is beneficial for a broad array of software developers within the company: developers in the Labs, in Infrastructure/Functions, as well as product generation groups.

5 Conclusion

We've defined the concept of Corporate Source and started a pilot project to understand the applicability of such concepts to a corporation like HP. In the seven-eight months that the service has been operational, about two dozen projects have been submitted with an active daily hit rate. While this response is encouraging, we believe the full potential of Corporate Source is yet to be achieved.

Some learnings from our early experiments are:

- Adoption of Corporate Source (or any Open Source initiative, for that matter) is at its heart is more of a process of social – than technical – change. While it does require learning of new tools (e.g., CVS) and coding according to specific standards that may be somewhat different than use in any particular division, the key elements are leadership to draw people into using Corporate Source, and getting developers (and their management chain) comfortable with sharing code broadly across the company.
- It's a challenge to find an effective starting point regarding which software domain should be an initial focus from which to populate the repository. While this naturally should be an area of value to a broad range of developers, if it gets too broad it won't have real value to anyone's unique coding tasks.
- Using the word "module" somewhat generically, it's a challenge to strike the balance between module simplicity and utility. This age-old problem resurfaces with Corporate Source because there's a tradeoff between how complete a module might need to be for another developer to consider adopting it into his or her code stream, yet not so complicated with irrelevant features, APIs, or calling sequences that it would take more time to reconfigure the module than write it from scratch. Effective heuristics for this are developing in an evolutionary way.
- The authors hold the opinion that what's most important is that the methodologies of Open Source are appropriate for many uses within the corporation: product development, certainly; but in equal measure Open Source provides great value to developers who concentrate on the corporate computing infrastructure itself, as well as those who design and build internal decision support tools.
- We use only one version control system: Concurrent Version System (CVS) [2]. Some software developers are already proficient in using other version control systems. We believe developers have the inertia to learn and experiment with another version control system.

- Often projects already make available source code for their projects on their own web service. Such projects don't see the need or benefit of moving towards a Corporate Source repository.

We are working on systematically improving and learning the Corporate Source service using an empirical research process.

Availability

The Corporate Source software is available for download from
ftp://src.hpl.external.hp.com/pub/open/hpcs-0.1.tar.gz

Acknowledgements

We thank Elsa Durante for writing the user interface for Corporate Source, and Chi Huang for managing the service. In addition, we thank the following people for their contributions to the ideas and implementation of Corporate Source: Jeff Archie, Sharon Beach, Alison Chaiken, Dick Cowan, Umesh Dayal, George Foreman, Martin Griss, Fred Luiz, Rajeev Pandey, Scott Peterson, Eugenie Prime, Debbie Seys, and Carl Staelin.

6 References

[1] M. E. Fagan. Advances in software inspections. *IEEE Transactions on Software Engineering*, 12(7), July 1986.

[2] Fogel Karl. *Open Source Development with CVS.* CoriolisOpen Press, 1999.

[3] E. Raymond. The Cathedral and the Bazaar. See http://www.tuxedo.org/~esr/writings/cathedral-bazaar.