Configuration Management Workspace Awareness for Distributed Software Development

Anita Sarma
University of California, Irvine
127B Computer Science Trailer
Irvine, CA 92697-3430
1-949-824-5160
asarma@ics.uci.edu

ABSTRACT

Current configuration management systems isolate workspaces such that work in one workspace is shielded from parallel changes in other workspaces. This workspace isolation is needed so that changes in one workspace do not interfere with changes by another developer working in parallel. This isolation, however, creates a problem in that developers are not aware of other workspace activities. As a result, developers often end up making conflicting changes that on promotion to a central configuration management repository have to be resolved - a time consuming task. To address this problem we are building Palantír, a novel workspace awareness tool. Palantír raises awareness among developers by providing them with continuous information about concurrent changes and how these changes would affect their work. Palantír thus moves the point at which conflicts are detected earlier, from being only at the moment at which changes are committed to continuously during the making of changes. This allows early detection of potential conflicts and should result in fewer and smaller conflicts.

1. INTRODUCTION

To address the circumstances of parallel work and the presence of potential conflicts, modern configuration management systems typically follow one of two mechanisms [1]. The first allows only a single line of development by locking artifacts, such that there is no parallel development and hence no conflicts. The second mechanism allows parallel development, but conflicting changes have to be resolved before they are promoted to the central repository. In both these cases, however, workspace activities remain isolated and developers remain unaware of each others work. In essence, thus, there are two kinds of isolation that a configuration management system provides: *good isolation*, where work in workspaces is shielded from parallel activities to allow unhindered development, and *bad isolation*, where developers have no idea of parallel change. Good isolation is well supported by configuration management systems. Unfortunately, bad isolation is a side effect that often leads to significant development problems.

The root cause of the problem is that both pessimistic and optimistic approaches isolate workspaces and developers are not aware of parallel activities except when they explicitly interact with the repository. Even then, in the pessimistic approach a developer only knows what artifacts other developers have locked for future changes while in the optimistic approach they only become aware of changes that have already occurred. In essence, developers are not aware of parallel changes, nor are they provided any mechanism to detect them.

Palantír is a novel workspace awareness tool that retains the good isolation of configuration management systems, but breaks the bad isolation. Palantír provides developers with a continuous, up to date picture of all relevant changes in parallel workspaces. This gives developers the chance to become aware of other ongoing activities, and coordinate with each other or even adjust ones own activities based on this information, such that conflicts are minimized.

2. HYPOTHESIS

Palantír is based on the hypothesis that conflicts in parallel development can be considerably reduced, both in number and magnitude, by providing developers with insight into ongoing changes in parallel workspaces. We believe that, rather than solely relying on the configuration management system for coordination activities, a much better result can be achieved if developers are provided with the right information to self-coordinate their actions within the framework of the configuration management system. This observation is based upon much research in the CSCW (Computer-Supported Cooperative Work) field, where it has been concluded time and again that strict, codified processes leave little flexibility to developers and generally create an ineffective process [3,2].

Our work attempts to move away from strictly codified configuration management processes by providing developers with the mechanisms and information to self-coordinate. In doing so, workspaces still provide a private place to make changes, and the configuration management tools still provide mechanisms to merge changes and other functionalities, but information about all changes in all workspaces becomes public. This allows developers to spot conflicts as they emerge, upon which they can then decide that a conflict is small and can be neglected, or that it has the potential of becoming large and complicated (at which point they probably should contact the other developer before progressing with their own plans). By moving earlier the point of detection of conflicts from the commit stage to during actual changing of the artifacts, developers can thus reduce the number and magnitude of conflicts.

3. APPROACH

Palantír builds on top of existing configuration management facilities and concentrates on the collection, distribution, organization and presentation of relevant workspace information. As we do not want to build a new configuration management system altogether, we have architected Palantír as shown in Figure 1. The arrows represent information flow. Gray ovals are configuration management system components which are used unchanged; white ovals (except an event service that we reuse) represent components of Palantír.

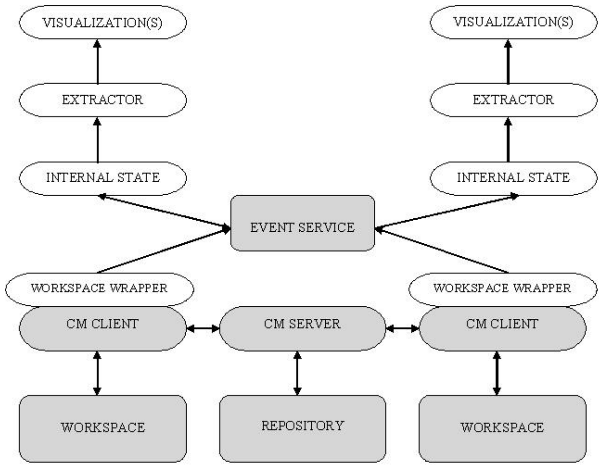

Figure 1. Palantír Architecture.

A *workspace wrapper* collects and translates configuration management specific activities to Palantír events, which are then distributed by the generic *event notification service* to other workspaces. The *internal state* stores the events, which are then organized and extracted by the *extractor* before being displayed by one or more *visualization* components. The internal state subscribes to events that concern actions on artifacts that are present in the current workspace, thus making Palantír more scalable. In our previous work [4], we have described Palantír in terms of its goals, architecture, and implementation.

4. CURRENT RESEARCH PLAN

Thus far, we have built a prototype with four different visualizations and implemented it with three configuration management systems. Next, we will concentrate on making the information more precise. Just knowing which artifacts have changed is not sufficient. We believe it would be helpful to know, for instance, how much has changed between the newly checked-in and previous version of an artifact. This allows a developer to be aware of the magnitude of the changes taking place. We call this measure the *severity* of changes. Different measures could be applicable at different times. We intend to explore the following measures: (1) calculating the relative number of *lines of code* that has changed, (2) detecting similarities based on *tokens* and replacing each keyword and variables with tokens before applying a differencing algorithm, and (3) analyzing the differences between abstract syntax trees of the two versions.

A second measure which we will explore is *change impact* analysis, which would provide the developer a synoptic view of how changes made by other developers affect their current work. We will explore the following measures: (1) calculating the relative number of overlapping *lines of code* that has changed, (2) calculate the relative number of *interfaces* that have changed, and (3) calculate the relative size of the dependency analysis graphs to determine the "reach" of a change. Different measures of severity and change impact may be suitable at different times. One of our goals is to find out which measures are more effective and more easily adopted by developers.

5. EVALUATION

We plan to evaluate Palantír in actual case studies. In particular, we wish to put Palantír in use with developers in their normal course of development and measure the number and magnitude of conflicts that occur as compared to previous development without Palantír. In addition, we will collect feedback on Palantír using questionnaires and personal interviews. We intend to run two case studies. First, we will run a case study with the current system as is, in order to determine if information on parallel activity alone helps developers in identifying potential conflicts. The second case study will be a final conclusive case study of Palantír as enhanced with advanced severity and change impact measures. The goal here is to establish which measures developers would use and find most effective in recognizing potential conflicts.

In case it is not possible to actually run case studies, our back-up plan is to simulate an actual development environment by running a configuration management archive "through" Palantír. Our goal is to determine whether concurrent changes that actually resulted in conflicts would have been indicated by Palantír, thereby demonstrating that Palantír would have had a positive effect in the real world of software development project.

6. CONCLUSIONS

Any software development project entails an inherent strain between the need for individual work and the need for the overall integration of individual changes into a final system. Current configuration management systems reduce this strain by providing workspace isolation. This isolation is both good and bad. On the one hand, individual work needs to be shielded from the effects of parallel changes. On the other hand, this isolates developers, which often results in concurrent changes that conflict. Additionally by sharing the severity and impact of these changes, Palantír provides developers with precise information to self coordinate and proactively avoid conflicts.

ACKNOWLEDGEMENT

Effort funded by the National Science Foundation under grant numbers CCR-0093489 and IIS-0205724.

REFERENCES

[1] Burrows, C. and I. Wesley, *Ovum Evaluates Configuration Management*. 1998, Burlington, Massachussetts: Ovum Ltd.

[2] Ellis, C.A., K. Keddara, and Rozenberg. *Dynamic change within workflow systems*. in *Proc of Conference on Organizational Computing Systems (COOCS'95)*. 1995. Milpitas, CA.

[3] Nutt, G.J. *The evolution toward flexible workflow systems*. in *Distributed Systems Engineering*. 1996.

[4] Sarma, A., Z. Noroozi, and A. van der Hoek. *Palantír: Raising Awareness among Configuration Management Workspaces*. in *Twentyfifth International Conference on Software Engineering*. 2003.