

Awareness for Distributed CM Workspaces

Motivation

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

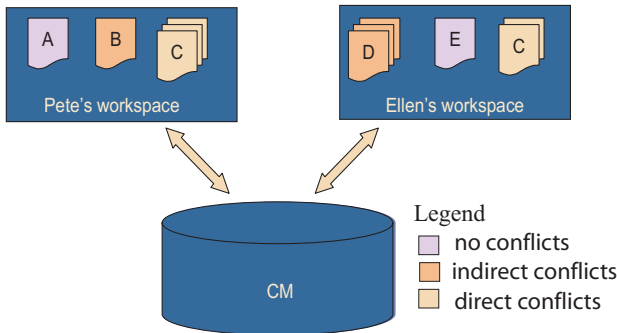


Figure 1. Example Workspace.

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. There are effectively two kinds of conflicts that can arise during parallel work. The first, direct conflicts, represent overlapping changes to the same artifact in different workspaces. For example, in Figure 1 both Ellen and Pete have locally modified artifact "C" in their workspace. While merge techniques may help to resolve such conflicts, more often than not manual resolution is needed. The second kind of conflicts, indirect conflicts, represent changes to an artifact in one workspace that negatively affect changes to another artifact in another workspace. In Figure 1, Ellen has modified artifact "D", which conflicts with changes that Pete has made to artifact "B". Unfortunately, existing configuration management systems do not provide any support for detecting and / or resolving indirect conflicts.

Palantír

Palantír is a novel workspace awareness tool that breaks workspace isolation in configuration management systems in order to better inform developers of ongoing activities in other workspaces. Palantír does so without breaking good isolation; developers still remain insulated from other changes being made in parallel workspaces. Bad isolation, however, is eliminated by Palantír: developers are continuously informed with an up-to-date picture of relevant changes being made in parallel workspaces. This allows developers to become aware of potential conflicts as they develop, and allows them to take proactive steps to minimize those conflicts. The point at which conflicts are detected, thus, is moved earlier, from being only at the moment at which changes are committed, to continuously during the making of the changes. As a result, fewer and less significant conflicts result, thereby shortening development time and reducing development cost.

A key characteristic of Palantír is that it not only shows which artifacts are changing, but also the severity ("how much has changed") and impact ("how much does that change influence the artifacts in my workspace") of each change. This information helps developers in gauging the seriousness of each conflict as

well as determining whether they need to undertake any actions at this point. For instance, using simple severity and change impact metrics, a set of changes could have a severity of 30% (e.g., the developer changes 30 out of a 100 lines of code), but the change impact could be 0%, since the developer did not change the interface of the code). Alternatively, it is possible to have a severity of 10% (developer only changed 10 out of a 100 lines of code), but the change impact could be 80% since the developer changes 8 of the 10 public functions.

Approach

Palantír builds upon existing CM facilities and concentrates on the collection, distribution, organization and presentation of relevant workspace information. The architecture of Palantír is shown in Figure 2. The arrows represent information flow. Blue ovals are configuration management system components which are used unchanged; Cream ovals represent components of Palantír. A workspace wrapper collects and translates CM 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.

Currently, Palantír has been integrated with RCS, CVS, and Subversion.

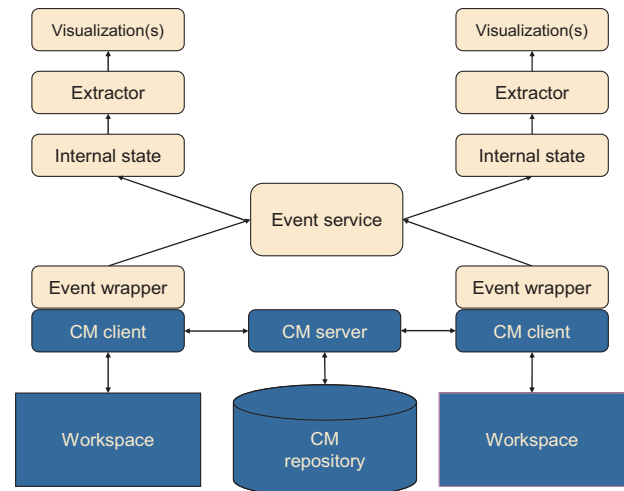


Figure 2. Palantír Architecture.

Key Characteristics

- **Non-obtrusiveness:** Palantír ensures that developers do not have to change the way they interact with their configuration management system. Simple workspace wrappers translate actions of developers to Palantír events, which are interpreted by the internal state component.
- **Scalability:** Informing a developer, of all parallel activities in all workspaces, has the potential to overwhelm the user and prove more detrimental than helpful. We leverage the event filtering mechanism of Siena (a generic event notification service) to only inform developers of relevant activities in other workspaces.
- **Flexibility:** Different developers and configuration management systems need different amounts of information for different levels of awareness. Therefore Palantír provides different visualizations, each detailing a different amount and detail of information.

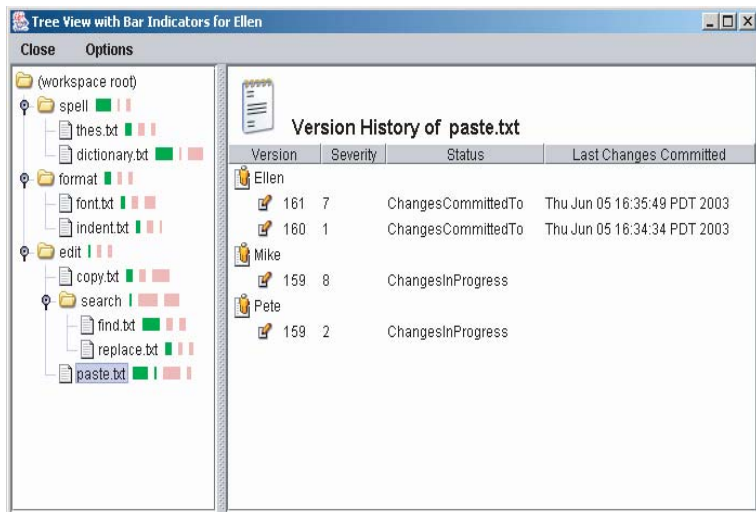


Figure 3. Tree Visualization.

- **Configurability:** It might not be always desirable to view all activities in all workspaces. Most of the time, a developer is interested in parallel activity on certain artifacts or by certain developers. The visualizations can be configured such that only a subset of desired events is viewed from the entire set.

Visualizations

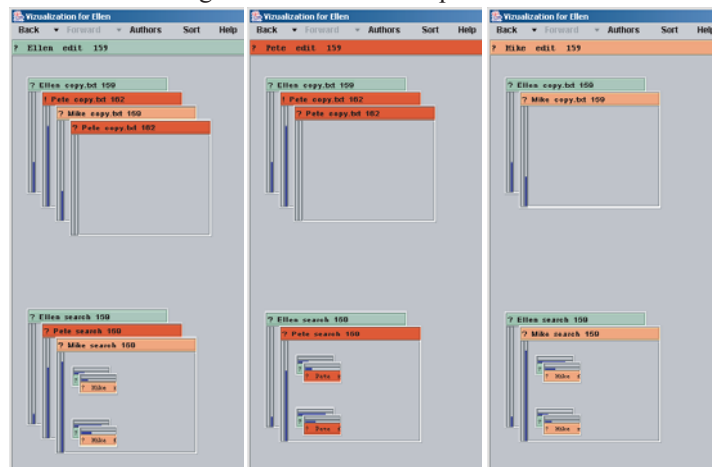
Different visualization components are responsible for organizing and displaying activities as they occur in the various workspaces. They primarily show which artifacts are being changed by which other developers, as well as the severity and impact of those changes. Thus far, we have developed four visualizations: (1) a ticker tape visualization, (2) a tabular visualization, (3) an explorer visualization, and (4) a fully graphical visualization. Each of these visualizations provides different amounts of information, presented in a different format. Developers can choose the visualization that best suits their need, or choose to use more than one in parallel.

Figure 3 shows the explorer visualization. The left hand side of the view is an expandable tree view, which is enhanced with vertical bars that indicate the severity of ongoing and committed changes: the longer the bar, the larger the size of the change. Changes are color coded to distinguish changes in a local workspace (green) from changes in other workspaces (red). Clicking on the name of an artifact presents the history of that artifact in each of the workspaces. The status of each version is listed (changes are in progress or changes have been committed) along with a severity and change impact measure. For instance, "paste.txt" is present in three workspaces; it has been committed twice in the first workspace; and it is in the process of being changed in the other two workspaces. This helps a developer in gauging whether they should contact any of the other developer to enter into a discussion to avoid future conflicts.

Figure 4 shows the fully graphical visualization that presents a developer with a hierarchical view of an artifact and its constituents (in this case version 159 of the folder "edit"). Each constituent artifact may itself contain other artifacts and each artifact in the view may exhibit multiple versions (as indicated by stacks of artifacts). Color coding separates different workspaces. For instance, the stack for the artifact "/edit/copy" indicates that Ellen, Pete, and Mike each have a version of the artifact in their workspace. Ellen and Mike each have version 159 in their workspace, and their changes are still in progress as indicated by the question mark. Mike, on the other hand, already has checked in a new version of the artifact (as indicated by the exclamation mark), resulting in his having version 162 in his workspace. Severity is indicated by the progress bar: the fuller the bar, the higher the severity.

A key feature of the fully graphical visualization is that it can show pair-wise conflicts. Simply by clicking on an artifact being changed or present in another workspace brings up a view that only includes local changes and the changes in that other workspace.

Figure 4. Pair-wise comparison.



Conclusions

Palantir is a workspace awareness tool that deliberately, but non-intrusively breaks CM workspace isolation by continuously informing developers of the activities of others developers. By letting developers know who modifies which artifacts and by how much, Palantir complements current automated procedures with the capability of human intervention when potential problems are recognized. The resulting system increases awareness among developers, and helps them in proactively coordinating their tasks such that future integration problems can be avoided.

References

Sarma, A., Z. Noroozi, and A.v.d. Hoek. Palantir: Raising Awareness among Configuration Management Workspaces, in Proceedings of the twenty fifth International Conference on Software Engineering. 2003, Portland, Oregon.

Contact Information

Professor André van der Hoek
Anita Sarma

Institute for Software Research
University of California
Irvine, California 92697-3425

For more information about this work
please visit the project website:

<http://www.ics.uci.edu/~asarma/Palantir>

or contact André van der Hoek at:

andre@ics.uci.edu
949-824-6326
949-824-1715 (fax)

Effort funded by the National Science Foundation under grant numbers CCR-0093489 and IIS-0205724. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Science Foundation.