Vermeidung von Merge-Konflikten durch Koordination- und Kommunikationsunterstützung in Eclipse

Workspace Awareness mit ceno

>> Andre Albert

„Je früher ein Konflikt erkannt wird, desto einfacher damit kostengünstiger kann er gelöst werden.“ Dieses Credo gilt besonders bei Erstellung einer Software in einem Entwickler Team. Durch fehlende Absprachen oder unzureichende Koordination kann es jedoch zu Merge-Konflikten bei der Synchronisation mit einem zentralen Code Repository kommen. Das ceno Plugin bietet einen Ansatz zur Vermeidung solcher Szenarien.

„Hat jemand von euch gerade Datei X geöffnet“ solche oder ähnliche Sätze kann man oft im Projektalltag eines Entwicklungsteams hören. Ist doch die Gefahr der gleichzeitigen Bearbeitung einer Datei mit zunehmender Anzahl von Entwicklern recht wahrscheinlich. Dies betrifft besonders Dokumente, die nicht im vollen Umfang einer bestimmten Aufgabe zuzuordnen sind und von zentraler Bedeutung sind. Ein Beispiel hierfür können unter anderem Konfigurationsdateien oder Dokumentationen sein. Auswirkungen einer solchen fehlenden Koordination können Merge-Konflikte beim synchronisieren mit einem zentralen Code Repository sein. Hierdurch entsteht in der Regel Mehraufwand, der je nach Art der Überschneidungen nur manuell auflösbar ist. Die Situation verschärft sich, sollte es sich um nicht textbasierte Dokumente wie Binärdateien oder Dateien, die mittels visuellem Editor erstellt wurden, handeln. Hier ist ein Merging oft nicht mehr sinnvoll oder gar unmöglich.

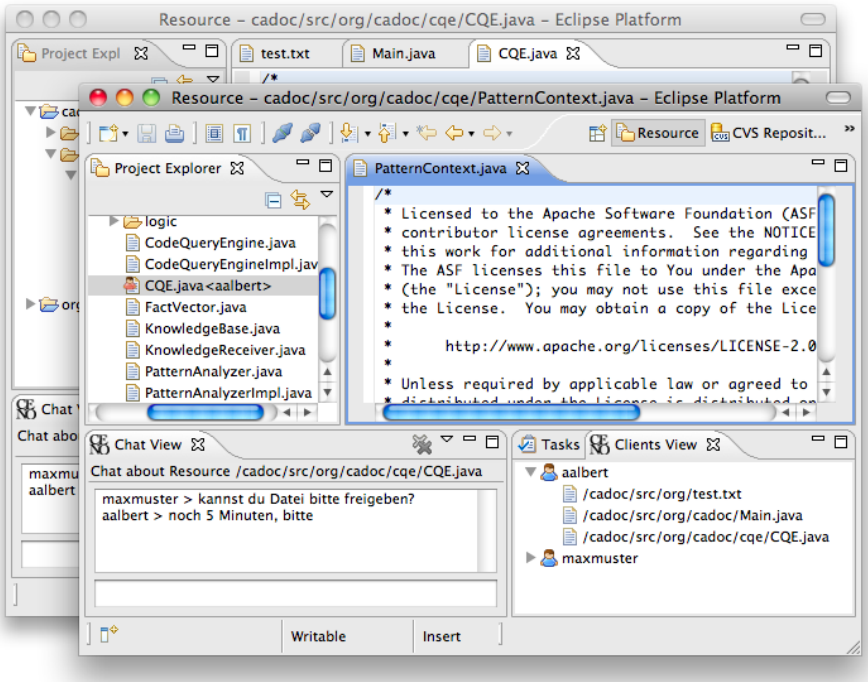
Eclipse bietet mit Modulen für Task und Versionsverwaltung bereits Unterstützung für die Arbeit in einer Gruppe an. Jedoch kann echtes Adhoc Wissen über die Aktivitäten der anderen Entwickler nur bedingt vermittelt werden. Diese Informationen werden erst bei einer Interaktion mit dem Repository ersichtlich - letztendlich zu spät für die Vermeidung des Konfliktes.

Die Herausforderungen bei der computergestützten Zusammenarbeit sind bereits seit den 80ern Gegenstand der Forschung. Unter dem Forschungsfeld CSCW (Akronym für Computer Supported Cooperative Work) werden Theorien für die Entwicklung von Groupware erforscht. Eine Schlüsselrolle wird dabei dem Wissen über die momentanen Aktionen und Aufgabenfelder der Teammitglieder zugesprochen[1]. Dieses Wissen wird in der Fachsprache als Awareness bezeichnet und lässt sich je nach Bezug unterschiedlich klassifizieren. Von besonderer Bedeutung mit Hinblick auf das eingangs formulierte Problemszenario ist der Bereich der Workspace Awareness. Dabei handelt es sich um Informationen über die Interaktion der anderen Benutzerinnen und Benutzer mit dem gemeinsamen Arbeitsbereich und den enthaltenen Artefakten (wikipedia). Mit Projekten wie Palantír wurden ersten Prototypen an der Universität von Kalifornien, Irvine[2] implementiert. Das in den Veröffentlichungen beschriebene Eclipse Plugin war nach unseren Recherchen jedoch nicht auffindbar. Die Projektseite ist nicht mehr erreichbar und eine letze Aktivität wurde laut 'wayback machine' des internet archives[3] im Januar 2006 gemessen.
Neben Implementierungen aus dem Forschungsumfeld bietet das Jazz Projekt, insbesondere der Rational Team Concert[4], eine Umgebung, die vergleichbare Features beinhaltet. In der an die Musik angelehnten Terminologie des Projektes spricht man hier von einer Concert Awareness. Dabei werden uncommitted Changes anderen Teammitglieder durch einen dekorierten Eclipse Package Explorer sichtbar gemacht[5]. Die Jazz Plattform geht weit über die Möglichkeiten hinaus und richtet sich mit Funktionalitäten zur Verwaltung von Softwarelebenszyklen und Einbeziehung verschiedener Stakeholder Rollen an eine andere Zielgruppe. Je nach Art der Projektgröße und gewerblichen Ausrichtung fallen jedoch Lizenzgebühren an. Auch ist der initiale Konfigurationsaufwand zum Aufsetzen der Jazz Plattform nicht zu unterschätzen.

Ceno Plugin

Mit dem ceno Projekt[6] gibt es seit einem Jahr eine Open Source Implementierung[7], welche die Ideen von Workspace Awareness umsetzt. Das ceno Plugin registriert sich als Listener für verschiedenste Workbench Aktionen und bietet Möglichkeiten, diese Events zu verteilen und auszuwerten. Das Plugin ist so-

1 Workspace Awareness for Groupware – Gutwin & Greenberg
2 http://www.isr.uci.edu/
3 http://www.archive.org/
4 http://jazz.net/projects/rational-team-concert/
5 Jazzing up Eclipse with Collaborative Tools – Cheng, Hupfer, Ross, Patterson – IBM Research
6 Http://ceno.sourceforge.net/
7 Eclipse Public License



mit in der Lage, den Zugriff auf Dokumente zu koordinieren.
So erscheint eine Warn-Meldung, wenn versucht wird eine Datei zu öffnen, welche bereits bei einem anderen Entwickler geöffnet ist. Warn-Meldungen können jedoch auf eigenes Risiko ignoriert werden. Neben dieser impliziten Wissensvermittlung erweitert ein eingebundener Decorator den Eclipse Package Explorer um Overlay-Icons oder auf Wunsch um die Namen der betroffenen Entwickler.
Basis für Kooperation ist neben Koordination die Kommunikation. Ein integriertes Messaging erlaubt es, alle Entwickler, welche eine bestimmtes Dokument geöffnet haben, anzuschreiben. So kann ein Interesse an der Arbeit an dem jeweiligen Dokument oder der Wunsch zum Checkin in das Repository geäussert werden. Ceno orientiert sich dabei an den bekannten Arten der Nachrichtenübermittlung:

- Unicast: Instant Messaging mit jeweils einem Teammitglied
- Multicast: Chat mit allen Entwicklern, die aktuell ein bestimmtes Dokument geöffnet haben
- Broadcast: Chat mit allen Mitgliedern des Projektteams

Bisher wurde vorrangig auf Echtzeit-Wissen eingegangen. Beteiligte Personen müssen also verbunden und aktiv sein. Was jedoch, wenn ein Entwickler an einen Dokument arbeitet und seine Änderung nicht in das Code-Repository eincheckt und sein Eclipse beendet?
Ereignisse müssen also über den Zeitraum einer Session persistiert werden. Erst nach einem späteren Checkin kann dieser Konflikt behandelt werden.
Dies zieht Auswirkung beim Entwurf der Anwendung mit sich. Eine Datenbank ist somit zentraler Bestandteil in einer Client-Server Architektur (Abb. 1). Im Stile eines Data-Warehouse Szenario werden alle Events der verbundenen Entwickler in einer Fakten-Tabelle abgespeichert und für Analyseanfragen bereitstellt. Hierbei kommt eine vergleichsweise leichtgewichtige dateibasierte Apache Derby Datenbank zum Einsatz. Der ceno Server lässt sich innerhalb der Eclipse Workbench starten und stoppen. Alternativ ist es auch möglich, den Dienst auf einem dedizierten Host über .sh oder .bat Skript starten.

Aus technologischer Sicht ist die Kommunikation verteilter Eclipse Workbench Instanzen interessant. Das entsprechende Protokoll ist innerhalb des Communication Layer (Abb. 1) implementiert und gekapselt. Basis derzeitiger Implementierung ist die Net4J Signal Plattform[8], da sich Workbench Events, wie das Öffnen einer Datei, gut durch Net4J Signale abbilden lassen. Entwicklern, welche sich bereits mit dem Eclipse Modeling Ökosystem befasst haben, ist das Projekt eventuell als Basis von CDO (Connected Data Objects)[9] bekannt.

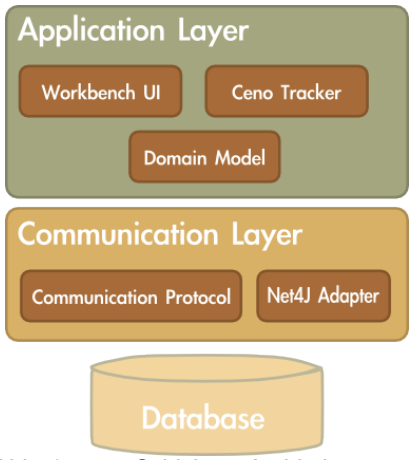


Abb. 1: ceno Schichten-Architektur

Aufbauend auf dem Communication Layer befindet sich fachliche Logik innerhalb des ceno Tracker Plugin. Dieses ist verantwortlich für das Abfangen, Absenden und Auswerten von Workbench Events. Es kann gewissermaßen als Schnittstelle zwischen ceno und Eclipse angesehen werden. So registriert es sich als Listener für zahlreiche Workbench Events und bedient sich der bekannten Eclipse Extension Points, um Contributions (Views, Commands, LabelDecorators …) an die Eclipse Plattform vorzunehmen.

Ausblick

Ceno ist ein Event-basiertes System, welches wiederum über Plug-ins erweitert werden kann. Neue Event-Handler können jederzeit

8 http://wiki.eclipse.org/Net4j
9 http://wiki.eclipse.org/CDO

über Extension Points registriert werden. Die Grenze für das Wissen über die Aktivitäten der Teammitglieder ist somit nur bedingt technischer Natur. Ein wichtiger Aspekt hierbei ist vielmehr der Datenschutz. So gilt es zu vermeiden, dass die Plattform zur Leistungsüberwachung der Entwickler missbraucht wird. So ist es (durch eigene Erweiterungen) theoretisch möglich, die benötigte Zeit die für das Erstellen eines bestimmten Dokumentes zu messen. Die zukünftige Entwicklung der Plattform wird verstärkt auf diese Problemstellung eingehen. So wird derzeit verstärkt an Filtermöglichkeiten gearbeitet, mit denen der Entwickler einstellen kann, welche Informationen er veröffentlichen möchte.

Fazit

Durch frühzeitige Vermeidung von Merge-Konflikten kann viel Zeit und Aufwand gespart werden. Basierend auf den Ergebnissen innerhalb Groupware und CSCW Forschung bietet das ceno Plugin eine freie und offene Implementierung für die Eclipse Plattform an. Durch Möglichkeiten zur Koordination und Kommunikation soll die Zusammenarbeit in einem Team von Entwicklern unterstützt werden.

Noch nicht alle erwähnten Punkte sind derzeit implementiert. Wir würden uns freuen neue Entwickler begrüßen zu können. Weitere Informationen zum Projekt findet man auf der Webseite:
http://ceno.sourceforge.net

Andre Albert ist IT Consultant bei der PRODYNA AG in Frankfurt am Main und arbeitet als Software-Engineer in RCP und Java-EE-basierten Projekten. Er ist Mitbegründer und Committer des ceno Plugins.