Saros: Eine Eclipse-Erweiterung zur verteilten Paarprogrammierung

Riad Djemili, Christopher Oezbek, Stephan Salinger

Freie Universität Berlin
Institut für Informatik
Takustr. 9, 14195 Berlin, Germany
{djemili, oezbek, salinger}@inf.fu-berlin.de

Abstract: Die im folgenden vorgestellte Diplomarbeit legt den theoretischen Rahmen für die verteilte Paarprogrammierung als Weiterentwicklung der klassischen Paarprogrammierung und beschreibt die Implementierung eines entsprechenden Eclipse-Plugins. Ein besonderes Augenmerk lag darin sinnvolle und realistische Anforderungen zu erarbeiten. Die Daten der Literaturanalyse wurden dazu mittels einer eigenen Umfrage unter Entwicklern ergänzt. Die Arbeit wurde von Riad Djemili durchgeführt und von Christopher Oezbek und Stephan Salinger als Forschungsprojekt der Arbeitsgruppe Software-Engineering am Institut für Informatik der Freien Universität Berlin betreut.

1 Paarprogrammierung

Paarprogrammierung (PP) bezeichnet eine Arbeitstechnik, bei der zwei Programmierer an einem Computer gemeinsame Artefakt (meist Code) bearbeiten [WKCJ00]. Man unterscheidet dabei zwei Rollen: Der *Driver* bearbeitet die Artefakte aktiv mit Maus und Tastatur, während der *Observer* die Eingaben kontrolliert und bei Entwurfsentscheidungen berät. Im Verlauf einer Sitzung kann die Rollenverteilung mehrmals wechseln. PP ist vor allem als Praktik des Extreme Programming (XP) bekannt geworden [Bec99].

Ziele von PP sind die Defektreduzierung, verbesserter Entwurf, Wissensaustausch und schnellere Produktentwicklung. Kritisiert wird jedoch häufig der Kostenaspekt und der Bedarf an räumlicher Nähe und geräumigen Arbeitsplätzen [Nos98]. Befürworter der PP betonen hingegen, dass die eventuell höheren Kosten vor allem durch die verbesserte Codequalität, mehr als kompensiert werden [WKCJ00].

2 Verteilte Paarprogrammierung

Bei der verteilten Paarprogrammierung (engl. distributed pair programming, DPP) findet die gleichzeitige und gemeinsame Bearbeitung des Artefakts von verschiedenen Ar-

beitsplätzen aus statt [SS01]. Diese allgemeine Definition ermöglicht unterschiedliche Umsetzungen in der Praxis: Müssen nur Textänderungen übertragen werden oder auch Dateioperationen? Ist die Übertragung von Gestiken und Gesichtsausdrücken per Video notwendig? Sollte der Observer sich vom Sichtbereich des Drivers lösen dürfen?

Ein Vorteil von DPP ist die Unterstützung virtueller Teams, welche Software-Entwicklung bei räumlicher Trennung betreiben und in erster Linie mittels elektronischer Medien kommunizieren. In der Open-Source-Entwicklung stellen diese Teams den Normalfall dar. Während bei PP der Driver den Aufmerksamkeitsbereich des Paares bestimmt, identifizieren wir bei verteilter Paarprogrammierung drei Grade von Parallelität:

Parallelität auf Programmebene: Der Observer kann das Fenster des DPP-Werkzeugs in den Hintergrund rücken, um mit anderen Programmen zu interagieren.

Parallelität auf Sichtebene: Der Observer kann von dem Aufmerksamkeitsbereich des Drivers abweichen und eigenständig im gemeinsamen Projekt lesen.

Parallelität auf Schreibebene: Es gibt mehr als einen Driver, das heißt mehrere Personen können gleichzeitig in dem Projekt schreiben.

Dabei ist zu beachten, dass mit steigender Parallelität die ursprünglich postulierten Vorteile der PP, z.B. die aus der stetigen Durchsicht resultierende Qualitätsverbesserung, verloren gehen können. Insofern bleibt zu untersuchen, welcher Grad an Parallelität noch zu einer Effizienz- und Qualitätssteigerung beitragen kann und ab wann diese kontraproduktiv wirkt. Eine weitere Kritik an DPP richtet sich gegen die eingeschränkten Möglichkeiten für den Observer die Aktivitäten des Drivers nachzuvollziehen (engl. awareness), z.B. anhand von Sprache und Gestik.

3 Technische Werkzeugansätze

Der Forschungsbereich der Computer Supported Cooperative Work (CSCW) unterscheidet im Wesentlichen zwei Implementierungsansätze [Han05]. Beim *Desktop Sharing* wird die Ansicht einer Arbeitsfläche über ein Netzwerk auf den Schirm eines oder mehrerer anderer Systeme übertragen, häufig zum Zweck von Fernadministration oder Demonstrationen. Vorteilhaft an diesem Ansatz ist, dass Werkzeuge, die in der DPP-Sitzung verwendet werden, nicht speziell angepasst werden müssen. Der im Rahmen der Diplomarbeit favorisierter Ansatz ist die *Collaboration Awareness*, bei der die Mehrbenutzer-Unterstützung unmittelbar in das Werkzeug integriert wird. Von Vorteil ist hier, dass das Werkzeug durch das Verstehen des Kontextes unterstützende PP-Funktionalitäten anbieten kann (beispielsweise eine Historie der letzten Aktionen des Drivers) und zudem einen höheren Grad an Parallelität erlaubt. Zudem leidet dieser Ansatz nicht unter den technischen Restriktionen des Desktop Sharing: Es gibt keine Probleme mit abzustimmenden Auflösungen oder geringen Bildwiederholraten.

4 Umfrage

Um die Anforderungen weiter zu ergründen und zu priorisieren, wurde im Juli 2006 eine Online-Umfrage durchgeführt, welche über Webseiten zu den Themen PP und XP angekündigt wurde [Dje06]. Es nahmen 44 Personen an der Umfrage teil. Die meisten davon (23%) arbeiteten mit Java und verfügten über durchschnittlich 12 Jahre Berufserfahrung. Zur knappen Zusammenfassung der Ergebnisse unterteilen wie die Anforderungen in sehr wichtig, wichtig und weniger wichtig.

Während die Funktionalitäten Chat und Stimmübertragung als *sehr wichtig* eingestuft wurden, galten Webcams als *weniger wichtig*. In Bezug auf die Parallelität wurde die Sichtparallelität als *wichtig* eingestuft. Als *weniger wichtig* wurde die Schreibparallelität bewertet. Als *sehr wichtig* galt, dass der Observer Informationen über die aktuelle Tätigkeit des Drivers erhält. Der umgekehrte Fall wurde als *weniger wichtig* eingestuft.

Zusammenfassend bestätigte die Umfrage den zuvor gewählten Ansatz der Collaboration Awareness.

5 Saros

Ausgehend von diesen Grundlagen wurde Saros als eine DPP-Erweiterung für Eclipse entwickelt[1] [Dje06]. Sitzungen werden über eine Kontaktliste mit Instant-Messaging-Funktionalität eingeleitet. Saros unterstützt Sitzungen mit einem Driver und einem Observer. Die Textbearbeitung wird in Echtzeit übertragen und eine Sichtparallelität unterstützt. Abbildung 1 zeigt verschiedene Awareness-Funktionalitäten.

Um dem Observer kontextbezogene Hilfe, Übersetzungsfehler und Werkzeugunterstützung zur Navigation anzubieten, erfolgt eine Replikation aller relevanten Dateien eines Projekts und aller Vorgänge innerhalb der Sitzung. Dies schließt auch das Erzeugen und Löschen von Dateien und Verzeichnissen ein. Die Replikation hat den Vorteil, dass im Gegensatz zum Ansatz des virtuellen Zugriffs, bei dem der Compiler Abhängigkeiten in Netzwerkanfragen übersetzt, der Compiler nicht angepasst werden muss und der Editor ohne Verzögerung bedient werden kann.

Ein Nachteil dieses Ansatzes ist die nötige anfängliche Synchronisation des Projekts. Um diese zu minimieren, vergleicht der Klient des Empfänger einer Sitzungseinladung alle seine lokalen Projekte mit der Dateiliste des zu bearbeitenden Projekts. Das lokale Projekt mit der größten Übereinstimmung wird zur Minimierung der folgenden Synchronisation genutzt.

Saros wurde auf Basis des offenen XMPP-Protokolls (bekannt durch Jabber) realisiert. Durch die Erweiterbarkeit des Protokolls kann Saros auf den bereits existierenden XMPP-Serververbund zurückgreifen und braucht keine eigene Server-Infrastruktur. Es ist außerdem möglich, Saros mittels separater Plugins um zusätzliche Vorgänge zu erweitern.

[1] Saros ist unter GPL lizensiert und kann unter http://sf.net/projects/dpp bezogen werden.

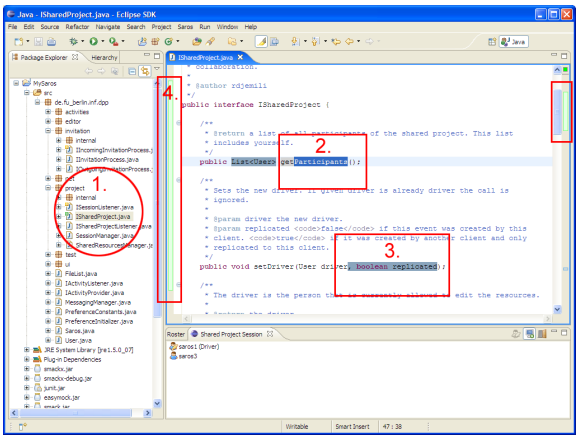

Abbildung 1: Ein Bildschirmfoto von Saros. Die Markierungen zeigen Awareness-Funktionalitäten: 1.) Dateien, welche der Driver gerade bearbeitet 2.) Aktuelle Cursor-Position bzw. Textauswahl des Drivers 3.) Textänderungen des Drivers 4.) Sichtbarer Edtiorausschnitt des Drivers.

6 Evaluation und Ausblick

Im Rahmen der Arbeit wurde eine einsatzfähige Version der Software fertiggestellt. Eine erste Evaluation mit zwei Programmierern an gegenüberliegenden Computern bestätigte, dass die Stimmkommunikation von essentieller Bedeutung ist. Gleichzeitiges Schreiben wurde dagegen nur anfags vermisst und Textauswahl häufig als Gestikersatz genutzt.

Um die Auswirkung der Parallelität bei DPP zu erforschen, ist als nächster Schritt eine Implementierung der Schreibparallelität, Verbesserung der Awareness und eine anschließende experimentelle Evaluation der verschiedenen Parallelitätsgrade geplant.

Literatur

[Bec99] K. Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.

[Dje06] R. Djemili. Entwicklung einer Eclipse-Erweiterung zur Realisierung und Protokollierung verteilter Paarprogrammierung. Diplomarbeit, Freie Universität Berlin, Inst. für Informatik, 2006.

[Han05] B. Hanks. *Empirical Studies of Distributed Pair Programming*. Dissertation, University of California, Santa Cruz, 2005.

[Nos98] J. T. Nosek. The case for collaborative programming. *ACM*, 41(3):105–108, 1998.

[SS01] T. Schümmer und J. Schümmer. Support for Distributed Teams in eXtreme Programming. In *eXtreme Programming Examined*. Addison Wesley, 2001.

[WKCJ00] L. Williams, R. R. Kessler, W. Cunningham und R. Jeffries. Strengthening the Case for Pair Programming. *IEEE Software*, 17(4):19–25, /2000.