Provisioning 2.0: Diffusion kleinteiliger Software in sozialen Netzwerken

Sebastian Draxler, Hendrik Sander, Gunnar Stevens

Lehrstuhl für Wirtschaftsinformatik und Neue Medien,
Universität Siegen
Hölderlinstrasse 3, 57068 Siegen
{sebastian.draxler|gunnar.stevens}@uni-siegen.de;
hendrik.sander@student.uni-siegen.de

1 Einleitung

Der Trend zu verteilter Projektarbeit, als auch der Trend zu kleinteiliger, kontinuierlich weiterentwickelter Software[1] verstärken die Notwendigkeit Softwarewerkzeuge schnell und flexibel in den lokalen Kontext von Arbeitsgruppen zu integrieren. Dies stellt neue Anforderungen an Provisioning-Werkzeuge (Wolf 2003; Gerlach, Güven et al. 2007; Reiswich 2008), welche die Bereitstellung, Konfiguration, Wartung und Administration von IT-Systemen in Unternehmen unterstützen sollen. Insbesondere gilt es die zumeist zentralistischen Ansätze um Formen des dezentralen, selbst-organisierten IT-Management zu ergänzen, um so die Aneignung und Diffusion von Werkzeugen und Werkzeugexpertise in agilen Projektteams zu unterstützen.

In diesem Beitrag stellen wir deshalb mit Peerclipse einen neuen Ansatz vor, der die Rolle der Kollegen als „Empfehlungssystem" für die Auswahl und Aneignung geeigneter Tools softwaretechnisch unterstützt und das Gruppenbewusstsein über Werkzeugnutzung im Team fördert.

[1] Beispiele solcher kleinteiliger Software stellen z.B. Microsoft Gadgets, iGoogle oder Widgets für das Apple Dashboard dar. Eine andere Klasse stellen die Erweiterungen (Plug-ins) für Standard-Produkte, wie z.B. Word, Skype, Firefox oder Eclipse, dar.

2 Softwareverteilung im Unternehmen

2.1 Bereitstellung durch zentrale IT-Abteilung

Die Bereitstellung, Konfiguration und Wartung von IT-Systemen in Unternehmen stellt eine zeitintensive und sicherheitskritische Aufgabe dar. Um das damit verbundene IT-Management zu systematisieren, beschreiben Standards wie ITIL[2] oder CobiT[3] die nötigen Aufgaben mithilfe eines ‚Best Practices' Ansatzes. Die Bereitstellung und Konfiguration von IT-Systemen ist dabei Teil allgemeiner IT-Dienstleistungen, die von sogenannten Service Providern durchgeführt werden. Der Service-Provider kann eine zentrale IT-Abteilung sein, die Dienstleistungen können aber auch ausgelagert werden (Buchsein, Victor et al. 2008).

Als Kunde einer IT-Dienstleistung wird dabei meist nicht der einzelne Nutzer betrachtet, sondern die Organisation als Ganzes. Zu der kooperativen Arbeit zwischen Nutzer und Service Provider findet sich im ITIL Standard entsprechend wenig. So werden beispielsweise kaum Aussagen zur Anpassbarkeit des eigenen Arbeitsplatzes gemacht. In der Tendenz zeigt sich aber, dass individuelle Anpassungsbedarfe nicht berücksichtigt werden, stattdessen wird versucht diese Aufgabe des Anpassens weitestgehend vom Nutzer fernzuhalten. Um die IT-Landschaft im Unternehmen überschaubar zu halten, sowie Kompatibilitätsprobleme bzgl. heterogener Bestandteile eines Systems zu vermeiden, führt die Umsetzung des Standards (zumeist implizit) zu einer Taylorisierung der Arbeitsplatzgestaltung.

Diese Tendenz reproduziert sich in kommerziellen Provisioning Lösungen, wie IBM Tivoli, bei denen nicht die Anpassbarkeit an individuelle Bedarfe des Nutzers im Vordergrund stehen. Vielmehr fokussieren die Lösungen auf die Bedarfe des Service Providers um Software möglichst zentral, automatisiert und synchron auf mehreren Arbeitsplätzen zu installieren, konfigurieren und zu warten (vgl. Coupaye und Estublier 2000; Gerlach, Güven et al. 2007; Reiswich 2008). Entsprechend fehlen Mechanismen zum Konfliktmanagement (Wulf 1997) welche eine effiziente Aushandlung der individuellen Gestaltung des Arbeitsplatzes erlauben. Zudem ignorieren zentralistische Ansätze die Bedeutung lokaler Netzwerke, welche – wie im nächsten Abschnitt dargelegt - als wichtiger Faktor bei der Diffusion und Aneignung von Software gelten.

2.2 Innovationsdiffusion in sozialen Systemen

Die Diffusion of Innovation (DOI) Theorie untersucht sowohl theoretisch, als auch empirisch Diffusionsprozesse von Innovationen innerhalb von sozialen Sys-

[2] IT Infrastructure Library (ITIL): Regel- und Definitionswerk zur Umsetzung einer IT-Infrastruktur und den notwendigen Prozessen, der Aufbauorganisation sowie den Werkzeugen.

[3] Control Objectives for Information Technology (CobiT): Framework zur IT-Governance, welches die Aufgaben der IT in Prozesse und "Control Objectives" gliedert.

temen (Rogers 2003). So wurden u.a. der Einfluss verschiedener Merkmale auf die Adoption von Innovationen herausgearbeitet und die Bedeutung lokaler Kommunikationskänale dargelegt (was auch für die Gestaltung von Provisioning Lösungen relevant ist):

"Mass media channels are more effective in creating knowledge of innovations, while interpersonal channels are more effective in forming and changing attitudes toward an innovation and thus in influencing the individual's decision to adopt or reject the innovation. Most individuals evaluate an innovation, not on the basis of scientific research by experts, but on the basis of the subjective evaluations of near peers who have already adopted the innovation. These peers serve as models whose behavior is imitated by others in the social system." (Rogers 2003).

Des Weiteren wurde in der DOI eine Differenzierung hinsichtlich der Innovationsbereitschaft (innovativeness) der Personen eines sozialen Systems vorgenommen. In verschiedenen Studien hat sich gezeigt, dass der Zeitpunkt der Aneignung mit individuellen Eigenschaften einer Person, aber auch mit der Art seiner Vernetzung in der jeweiligen lokalen Community korreliert. Die von der DOI vorgeschlagene Einteilung der Beteiligten in *Innovators (2,5%), Early Adoptors (13,5%), Early Majority (34%), Late Majority (34%)* und *Laggards (16%)* weist dabei Ähnlichkeiten zu Klassifikationsschemata auf, die unabhängig von der DOI in der CSCW-Forschung vorgeschlagen worden sind. Die für die Gestaltung von kooperativen Provisioning Lösungen interessantesten Typen sind dabei:

- *Innovators*: Innovatoren zeigen großes Interesse für das Neue und übernehmen solche Entwicklungen als erste. Sie nehmen in dem sozialen System eine Außenseiterrolle ein und pflegen eher soziale Kontakte zu anderen Innovatoren innerhalb und außerhalb des jeweiligen sozialen Systems.
- *Early Adaptors*: Sie sind stärker in das soziale System integriert und haben die Meinungsführerschaft inne. Andere potenzielle Nutzer fragen sie um Rat und informieren sich bei ihnen über Innovationen. Sie fungieren so im sozialen System zumeist als Change-Agents.
- *Early Majority*: Sie übernehmen eine Innovation schneller als der Durchschnitt. Sie interagieren zwar viel mit den anderen Mitgliedern, haben aber selten eine Meinungsführerschaft inne.

Weiterhin beschreibt die DOI, dass in Organisationen die nach Rogers durch formale, hierarchische Strukturen festgelegt sind Innovationsentscheidungen von wenigen Individuen auf Grund ihrer Entscheidungsgewalt oder technischer Expertise getroffen werden. Diese Sichtweise stimmt mit der Sichtweise von IT-Service Management Standards überein (bei der die technische Expertise primär bei der zentralen IT-Abteilung verortet ist). Wie im nächsten Abschnitt dargelegt, hat demgegenüber die ethnographische CSCW-Forschung gezeigt, dass zwar durch formale Entscheidungsstrukturen der Diffusionsprozess überformt wird, nichtsdestotrotz die interpersonale Kommunikation zwischen Kollegen eine sehr wichtige Rolle bei der Aneignung von Software einnimmt.

2.3 Diffusion lokaler Anpassungen

Insbesondere in den 80er und 90er Jahren wurden in der CSCW-Forschung verschiedene empirische Studien bzgl. der sozialen Aspekte der Anpassung von Software-Systemen in Organisationen durchgeführt (siehe Mackay 1990; MacLean, Carter et al. 1990; Gantt und Nardi 1992; Wulf 1999; Kahler 2001). Im Gegensatz zur DOI Theorie fokussiert die CSCW-Forschung stärker auf die Diffusion lokaler Innovationen. Hierbei zeigten sich bei der Diffusion von lokalen Innovationen in Organisationen ähnliche Muster wie beim Diffusionsprozess in anderen sozialen Systemen. So findet sich z.B. bei Mackay (1990) ein ähnliches Klassifikationsschema wie bei der DOI, wobei sie nur drei Gruppen unterscheidet:

- Die kleinere Gruppe der *Lead Users of new Technology* (ähnlich zum Typ des *Innovators*). Sie setzen sich intensiv mit neuer Software auseinander, nehmen Anpassungen vor und stellen sie Anderen zur Verfügung.
- Die Gruppe der *Translators* (ähnlich zum Typ des *Early Adoptors*). Sie besteht aus weniger technisch versierten Personen. Sie vermitteln zwischen der Gruppe der Lead User und normalen Nutzern, wobei sie die Anpassungen der „Lead User" interpretieren und an die individuellen Bedürfnisse der „User" anpassen.
- Die Gruppe der normalen *User (*dies ist eine Residualkategorie, welche *Early Majority, Late Majority* und *Laggards* in sich vereinigt). Sie passen selbst nicht an, sondern übernehmen Anpassungen primär von Anderen.

In ihrer Studie zum kollaborativen Anpassungsverhalten von CAD-Systemnutzern identifizierten Gantt und Nardi (1992) ähnliche Rollen, wobei sie zwischen *End User*, *Local Developer* (entspricht dem *Translator* bei Mackay) und *Professional Programmers* unterscheiden. Insbesondere heben sie hervor, dass End User in der Kooperation mit Anderen wesentlich komplexere Anpassungen vornehmen können, da sie so von erfahrenen Nutzern Anpassungen übernehmen können und in der Kooperation technische Expertise erwerben. Ähnliches zeigt sich auch in der Studie von MacLean et al. (1990). Darüberhinaus zeigten Gantt und Nardi (1992), dass in manchen Unternehmen engagierten Nutzern die Rolle des *Local Developers* offiziell übertragen wurde. Der Vorteil von *Local Developers* wurde dabei darin gesehen, dass sie mit der lokalen Arbeitspraxis vertraut sind und so die informelle Seite kooperativer Aneignung von IT-Systemen besser unterstützen können, als dies von einem ausgelagerten IT-Service Provider möglich wäre.

2.4 Die stille Leistung kooperativer Aneignung

Die Aneignung von IT-Systemen ist zum großen Teil ein informeller Prozess der von Akteuren als solcher zumeist gar nicht wahrgenommen wird. Aneignung ist deshalb zumeist eine ‚stille Arbeitsleistung' (Bolte and Porschen 2006), bei der formale Strukturen situativ unterfüttert werden. Neben dem Anpassen des IT-Systems an den lokalen Kontext stellt das informelle, explorierende Lernen einen

wichtigen Bestandteil dar (vgl. Pipek, Stevens et al. 2008). So merken auch George et al. (1995) an: *"People learn about computing during the course of adopting, altering, and using it in their work"*. Die soziale Dimension wurde dabei von Twidale unter dem Begriff des „Over the Shoulder Learning" (OTSL) untersucht: *"Over The Shoulder Learning is the informal, spontaneous workplace help-giving interaction that is often used by people to learn from their colleagues how to use part of a computer application."* (Twidale 1999). Das OTSL ergänzt dabei die Forschung zur Diffusion von Artefakten (Rogers, 2003) und die Forschung zum kooperativen Anpassen an den lokalen Kontext (Mackay 1990) um den Aspekt der Ausbildung lokaler Nutzungspraktiken als einen wichtigen Bestandteil situierter Innovationsprozesse. Dabei weist die Forschung zu OTSL auf die Bedeutung der gemeinsamen Arbeitspraxis für die Diffusion von Nutzungspraktiken hin. Durch die gemeinsame Praxis werden sowohl Gelegenheiten als auch ein gemeinsamer Kontext geschaffen, um Wissenstransfer zu erleichtern. Insbesondere zeigt die Forschung zu OTSL, dass IT-Systeme nicht Werkzeug-, sondern Aufgaben-zentriert angeeignet werden. Dies kann durchaus verschiedene (vom IT-Service Provider nicht antizipierte) Werkzeuge umfassen, welche zur Erledigung einer Aufgabe wichtig sind. Des Weiteren hat Twidale gezeigt, dass das vermittelte Wissen nicht allein auf die abstrakte Verwendung der Werkzeuge bezogen ist, sondern auch deren organisationale Einbettung umfasst.

2.5 Zusammenfassung

Im Unternehmen ist die Diffusion und Aneignung von IT-Systemen durch formale, hierarchische Strukturen geprägt. Die vergleichende Betrachtung der unterschiedlichen Forschungszweige zeigt jedoch, dass auch in solchen Fällen lokale Experten (bzw. *Lead User, Translator* oder *Local Developer)* für die effektive Nutzung von für den Arbeitskontext angepassten Werkzeugen eine wichtige Rolle spielen (siehe auch Won and Wulf 2003; Budweg, Stevens et al. 2008) und dass die Diffusion von Werkzeug und Werkzeug-Expertise miteinander verkoppelt ist. Trotzdem haben die Erkenntnisse zur informellen, selbst-organisierten Arbeitsplatzgestaltung nur vereinzelt Eingang in die Gestaltung technischer Unterstützungssystme gefunden (vgl. Kahler 2001; Won und Wulf 2003). Insbesondere existierende Provisioning-Ansätze konzentrieren sich allein auf formal erfassbare Aspekte der Bereitstellung von IT-Systemen. Demgegenüber werden die informellen Formen der Aneignung in lokalen Netzwerken zu wenig unterstützt. Die Provisioning Lösungen der Zukunft sollten daher um diese Aspekte ergänzt werden.
Darüber hinaus hat der Markt an frei verfügbarer, kleinteiliger Software in den letzten Jahren stark zugenommen, so dass der Nutzer sich häufig entscheiden muss, ob es günstiger ist eine passende Lösung im Netz zu suchen, oder eine bestehende Anwendung anzupassen, d.h. er steht immer häufiger vor der Frage: *adopt or adapt*? Als Folge davon bedarf es einer vergleichenden Betrachtung der Adoptions- und der Adaptions-Forschung um dem Forschungsgegenstand gerecht zu werden.

3 Unterstützung der informellen Aneignung im Team

Im Rahmen des CoEUD Forschungsprojekts haben wir die informelle Aneignung kleinteiliger Software am Beispiel des Komponenten-basierten Systems Eclipse untersucht, für das ein Markt von mehren tausenden, frei verfügbaren Erweiterungen existiert. In unseren Untersuchungen zur Aneignung von Eclipse in Unternehmen (Schwartz 2007; Draxler, Sander et al. 2008a) zeigte sich, dass die in der Literatur allgemein beschrieben Aneignungs- und Diffusions-Muster im Fall von Eclipse auf die Verbreitung neuer Plug-ins, sowie deren Konfiguration und Nutzung angewendet werden können. Wie im nächsten Abschnitt genauer dargelegt, orientieren sich Nutzer bei der Zusammenstellung der eigenen Arbeitsumgebung zumeist an den Kollegen, während diese als „Empfehlungssysteme" für neue Werkzeuge fungieren. Zugleich hat sich aber auch gezeigt, dass solche Empfehlungen nicht systematisch erfolgen, sondern häufig auf zufälligen Ereignissen beruhen und diese Praktiken nur unzureichend software-technisch unterstützt werden.

3.1 Nutzung von Eclipse in der Praxis

Eclipse ist eine der meistgenutzten Arbeitsumgebungen in Software-Unternehmen. Die Nutzung dieser Umgebungen haben wir im Rahmen von CoEUD sowohl qualitativ, als auch quantitativ untersucht. Eine detaillierte Beschreibung der Studien findet sich in (Stevens and Draxler 2009), im Folgenden geben wir eine kurze Zusammenfassung. Die erste qualitative Studie wurde 2007 durchgeführt. Hier wurde die Eclipse-Nutzung in vier KMU durch teilnehmende Beobachtung und ergänzende semi-strukturierte Interviews untersucht. Diese Studie wurde 2009 wiederholt, wobei auch ein Großunternehmen und eine Forschungseinrichtung für Interviews gewonnen werden konnten. Zusätzlich haben wir 2008 eine quantitative

Tabelle 1: Umfang an Plug-ins in Eclipse basierten Arbeitsumgebungen (n= 76)

Anzahl unterschiedlicher Plug-ins im Sample	4944
Mimimale Anzahl an Plug-ins in einer Konfiguration	89
Maximale Anzahl an Plug-ins in einer Konfiguration	1008
Durchschnittliche Anzahl an Plug-ins pro Konfiguration	321

Studie durchgeführt, an der sich 136 Personen beteiligten und 76 Konfigurationen[4] beisteuerten (siehe Tabelle 1).

[4] Eine Konfiguration stellt hier eine von einem Benutzer konfigurierte Eclipse Installation dar. Anhand der Konfiguration lässt sich ablesen ob und inwiefern ein Benutzer die Installation angepasst hat.

In der Studie bestätigte sich das besondere Merkmal von Eclipse, eine Arbeitsumgebung bereitzustellen, in der diverse kleinteilige Software je nach den lokalen Bedarfen integriert werden kann[5]: *"Eclipse itself is basically a core platform on which extensions and plug-ins from any vendor can be added."* (EDC 2008).
In den verschiedenen Studien zeigte sich die hohe Diversität der Eclipse Konfigurationen. In der quantitativen Studie wurden im Sample keine zwei identischen Konfigurationen gefunden (vgl. Tabelle 1). In den qualitativen Studien fanden wir zum Teil große Unterschiede zwischen den Eclipse Konfigurationen, die von den Mitarbeitern innerhalb eines Unternehmens eingesetzt wurden. Die Analyse ergab weiterhin, dass eine Konfiguration im Schnitt in den letzten 80 Tagen verändert wurde. Dies zeigt, dass die Anpassung von Eclipse zwar gängige Praxis, aber keine tägliche Handlung ist (dies deckt sich mit der quantitativen Studie: 92,66% passen Eclipse an; davon 77,21% ab und zu). Der überwiegende Teil der Partizipierenden (71,32%) gab an, dass Eclipse auch von Kollegen genutzt wird. Dies bestätigt unsere Beobachtung, dass Eclipse häufig in Projektteams eingesetzt wird.

Weiterhin zeigten sich in der Aneignung neuer Plug-ins die in der DOI beschriebenen Muster. So sind die beiden wichtigsten Kanäle sich über mögliche Plug-in-Erweiterungen zu informieren, das Internet als Massenmedium (von 78,48% zu diesem Zweck genutzt) und die Kollegen als interpersonaler Kanal (von 54,43% zu diesem Zweck genutzt). Insbesondere gaben 75,44% an, dass sie schon einmal Plug-ins von Kollegen erhalten haben, entweder indem eine entsprechende URL weitergegeben wurde oder indem sie die Dateien direkt vom Kollegen kopiert haben (in 28% der Fälle). Die Praktik des Kopierens vereinfacht insbesondere die Aneignung kompletter Arbeitsumgebungen. Während das manuelle Kopieren einzelner Plug-ins auf Grund der Abhängigkeiten zu anderen Plug-ins mühsam und fehlerträchtig ist, wird die Aneignung kompletter Arbeitsumgebungen dagegen wird als einfacher, charmanter Weg gesehen: *„[D]ann kopiert man sich einfach [Eclipse]. Also ich hab der [C] bevor die halt anfangen muss sich ein Eclipse runterzuladen, dann Plug-ins suchen, hab ich einfach meinen Eclipseordner freigegeben, die hat den sich dann kopiert und dann, das ist ja das charmante eigentlich auch an Eclipse, dass man einfach das Verzeichnis kopieren kann und es läuft dann woanders […]."* (Nutzer A, Beta)

Weiterhin konnten wir in den qualitativen Studien häufig eine informelle Arbeitsteilung beobachten, die durch folgendes Zitat gut illustriert wird: *„Also ich muss ganz ehrlich sagen dadurch, dass die Kollegen alle da so weit vorne sind bin ich da sehr bequem und ich mach mich jetzt nicht auf die Suche nach einem Plug-in, was die anderen noch nicht kennen. Das würde ich wahrscheinlich eh nicht finden."* (Nutzer D, Unternehmen Gamma). Verallgemeinernd gesprochen weist die Praxis Eclipse um Plug-ins zu erweitern strukturell analoge Muster auf, wie wir sie aus der DOI-, der OTSL- und der CSCW-Forschung zum kooperativen Anpassen kennen.

[5] Im Gegensatz zu z.B. MS Visual Studio, dass als fertiges, schon konfiguriertes Produkt ausgeliefert wird.

In unseren Untersuchungen haben wir jedoch auch erkannt, dass die Anzeigung über Kollegen ad hoc geschieht und nicht systematisch durch die Komponenten-Verwaltung von Eclipse unterstützt wird.[6]

3.2 Design Studie: Peerclipse

Die Praxis der Eclipse Nutzung zeigt das Eclipse eine hoch-flexible Arbeitsumgebung mit einem aktiven Software Ökosystem ist. Auf dieser Grundlage haben wir einen Ansatz zur Förderung informeller Aneignung von Plug-ins im Team entwickelt. Um den Wissensaustausch über neue Plug-ins zu systematisieren ging es uns vor allem darum, dass Gruppenbewusstsein zu eingesetzten Werkzeugen entlang zweier Dimensionen zu fördern:

- **Werkzeug-Popularität:** Der Nutzer soll unterstützt werden, einen Überblick über die im Team benutzen Werkzeuge zu erhalten (also von Kollegen als angemessen und qualitativ hochwertig eingestufte).
- **Werkzeug-Expertise:** Darüber hinaus soll der Nutzer unterstützt werden, sich einen Überblick über die im Team vorhandene Werkzeug-Expertise zu verschaffen, und kompetente Ansprechpartner zu finden, die ihm bei der Aneignung neuer Plug-ins unterstützen können.

Im Peerclipse-Projekt haben wir eine prototypische Umsetzung des Konzepts entwickelt. Zur Förderung des Austauschs zwischen Kollegen realisiert Peerclipse ein Peer-to-Peer Netzwerk, mit dessen Hilfe die Arbeitsplatzgestaltung von Kollegen visualisiert und Werkzeugdiffusion gefördert wird. Hierzu wurde ein Plug-in implementiert, das in jede Eclipse Anwendung,[7] wie z.B. die Eclipse IDE oder Lotus Notes, integriert werden kann.

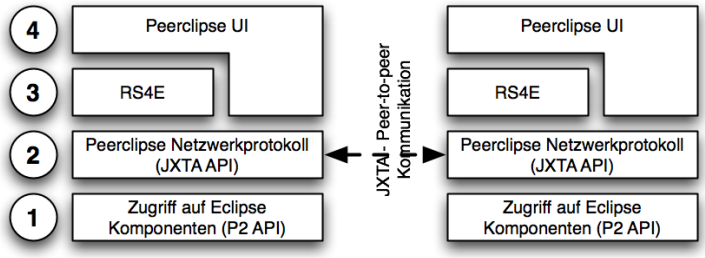

Abbildung 1: Peerclipse Komponenten Architektur

[6] So war bei Unternehmen Gamma ein Service-Provider vorhanden, der eine Eclipse Standard Installation für das Unternehmen bereitstellte. Diese wurde vom Team jedoch nicht genutzt, weil sie nicht auf die speziellen Bedarfe angepasst war.

[7] Unter http://www.eclipse.org/community/rcp.php findet sich eine Liste von Anwendungen, die auf Eclipse RCP aufsetzen.

Peerclipse wurde durch die in Abbildung 1 vereinfacht dargestellte Architektur realisiert: Auf unterster Ebene wird auf das Eclipse Provisioning-System *P2* aufgesetzt (Abbildung 1-1), mit dessen Hilfe die lokale Konfiguration analysiert und Plug-ins installiert bzw. deinstalliert werden. Insbesondere erlaubt *P2,* dass die lokale Instanz als Plug-in-Repository im Peer-to-Peer Netzwerk fungiert. Hierzu liest Peerclipse die nötige Information aus und stellt sie, unter Verwendung des von Sun entwickelten JXTA Peer-to-peer Protokolls, auf Anfrage anderer Peerclipse Instanzen zur Verfügung (Abbildung 1-2). Falls der Benutzer Plug-ins eines Kollegen installieren möchte, übernimmt Peerclipse die Aufgabe Abhängigkeiten bzw. Inkompatibilitäten zwischen den Plug-ins aufzulösen. Die so ermittelten Plug-ins werden vom lokalen Plug-in-Repository des Kollegen ausgelesen und über das Peer-to-Peer Netzwerk versendet. Dieses Verfahren ermöglicht es Peerclipse ad hoc und ohne Einrichtung eines zentralen Servers einzusetzen und lokale Gruppen-Repositories aufzubauen.

Die Informationen über die Konfiguration der Kollegen werden mit Hilfe von Recommender-Techniken (Klahold 2009) aufbereitet, um so populäre Werkzeuge zu identifizieren. Dieser Teil wurde in einem integrierten Recommender System für Eclipse (RS4E) gekapselt (Abbildung 1–3) und ist entsprechend leicht anpassbar. Die so aufbereiteten Informationen werden an der Benutzerschnittstelle visualisiert, um die Transparenz über die im Kollegenkreis benutzten Werkzeuge zu erhöhen und auf Grund der gesammelten Informationen Empfehlungen zu im Team genutzten Werkzeugen zu geben. Darüber hinaus wird diese Nutzungsinformation zur Vermittlung potenzieller Ansprechpartner und zur Erhöhung der Transparenz über die Werkzeug-Expertise im Team eingesetzt.

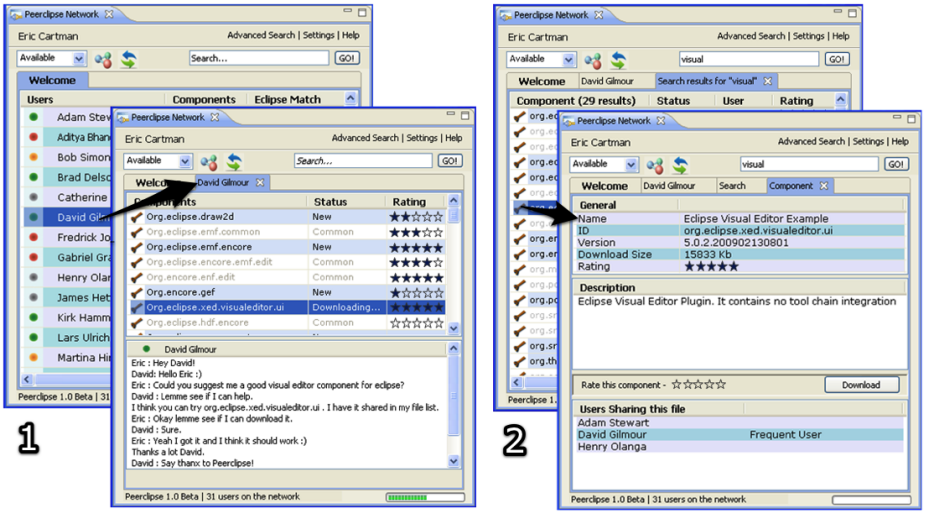

Abbildung 2: Die Benutzerschnittstelle Peerclipse User Interface.

Die Benutzerschnittstelle (Abbildung 2) wurde den Eclipse UI Guidelines folgend als eigene Eclipse View realisiert, um eine nahtlose Einpassung in die Arbeitsumgebung der Nutzer zu gewährleisten. Bei Bedarf zeigt die View eine Liste der Nutzer, die im Peer-to-Peer Netzwerk verfügbar sind. Interessiert sich der Nutzer nun für die Werkzeuge, die ein bestimmter Kollege einsetzt, markiert er diesen einfach (Abbildung 2–1). Dies öffnet einen neuen Karteikartenreiter, welcher die durch RS4E aufbereitete Information über die installierte Software des Kollegen anzeigt. Weiterhin kann der Nutzer hier Fragen zu den sichtbaren Plug-ins über den in Peerclipse integrierten Chat stellen (Abbildung 2–1 unten), was die Software zentrierte Kommunikation vereinfacht. Insbesondere können so auch auf eine einfache Art und Weise Konfigurationseinstellungen übermittelt werden, die z.B. per Telefon oder im direkten Gespräch schlecht weitergegeben werden können. Des Weiteren kann der Benutzer für ihn interessante Plug-ins markieren und per Kontext-Menü installieren. Dabei übernimmt Peerclipse das Berechnen des Plug-in-Abhängigkeitsgraphen, um ggf. zusätzlich benötigte Plug-ins mit zu übertragen und zu installieren. So vermeidet Peerclipse das fehlerträchtige manuelle Kopieren, als auch das umständliche Heraussuchen der Plug-ins im Internet.

Da selbst lokale Repositories sehr umfangreich sein können (siehe Tabelle 1) wurde eine Suche in Peerclipse integriert (Abbildung 2–2). Wird in der Suche ein Begriff eingegeben, durchsucht Peerclipse das Netzwerk nach Plug-ins, die mit dem Begriff in Verbindung stehen und zeigt die Treffer. Wählt der Nutzer einen Treffer aus, so erhält er Details zu diesem Plug-in und Informationen über andere Benutzer, die es einsetzen. So können potenzielle Ansprechpartner identifiziert werden, die bei Problemen mit diesem Werkzeug Hilfestellungen geben können. Über einen „Become an Expert"-Knopf können Benutzer interessante Plug-ins ihrer Arbeitsumgebung Anderen empfehlen. Die empfohlenen Plug-ins werden nach einem Ranking sortiert angezeigt, welches von RS4E aus der Anzahl der Benutzer und der Empfehlenden im Team berechnet wird. Wählt ein Nutzer nun solch ein Plug-in aus, werden ihm entsprechende Empfehler als potenzielle Ansprechpartner angezeigt.

3.3 Erste Erfahrungen

In einer ersten formativen Evaluation von Peerclipse haben wir für ein sechsköpfiges Team, welches Eclipse als primäre Arbeitsumgebung einsetzt, auf Grundlage der einzelnen Konfigurationen ein Gruppen-Repository erstellt. Zum einen zeigte sich wie in den Studien zuvor, dass die im Team eingesetzten Werkzeuge nicht identisch waren und bei der Rückspiegelung dieses Ergebnisses an die Gruppe waren die Teilnehmer überrascht was die Kollegen einsetzten.

Darüber hinaus interessierten wir uns für die Güte des in RS4E eingesetzten Verfahrens zur Aufbereitung der Information über das Gruppen-Repository. Daher wurden drei Varianten getestet. In der ersten Variante war das Gruppen-Repository zufällig sortiert. Die zweite Variante sortierte das Repository entlang

der meist genutzten Plug-ins. Variante drei individualisierte zusätzlich für den jeweiligen Benutzer das Repository. So wurden Plug-ins nicht angezeigt, wenn der Nutzer die aktuelle Version schon installiert hat. Umgekehrt wurden Plug-ins im Gruppen-Repository besonders markiert, wenn sie neuer waren, als das beim Nutzer installierte.

In der Evaluation zeigte sich, dass die Nutzer durchweg die individualisierte Variante präferierten. Insbesondere die Anzeige neuer Versionen stieß auf Zustimmung. Sie stellte eine wichtige Information dar, um sich im Team auf einen gleichen Stand zu verständigen und so eventuelle Konflikte zu vermeiden. Zudem wurde die Anzeige von selbst empfohlenen Plug-ins angeregt, um sich daran orientieren und mit anderen Experten austauschen zu können. In der Diskussion wurde auch die durch Peerclipse geschaffene Transparenz hervorgehoben. Auch die Anzeige geeigneter lokaler Ansprechpartner wurde als sehr interessant befunden. Besonders bei der Einschätzung von Empfehlungen zeigte sich, dass hier die Anzeige der Empfehlenden wichtig ist. Entsprechend der oben vorgestellten Rollenverteilungen nach Mackay (1990) wurden Empfehlungen von Personen im Team höher gewertet, wenn diesen allgemein eine höhere Werkzeug-Expertise zugewiesen wird. Dies ist ein Hinweis darauf, dass man dieser Rollenverteilung, bei der Verbesserung des Algorithmus eine höhere Gewichtung beimessen sollte.

4 Zusammenfassung

Die Verfügbarkeit von Plug-ins in weltweiten Software Ökosystemen wie Firefox, Skype oder Eclipse erreicht heute eine Komplexität, die sich deutlich von existierenden Studien wie Mackay (1990), MacLean et al. (1990) oder Kahler (2001) abhebt. Während die klassische CSCW-Forschung auf Gruppen immante Prozesse fokussiert, zeigen die Nutzungspraktiken von Eclipse wie lokale und globale Diffusions- und Adoptions-Prozesse miteinander verschränkt sind. Dadurch wird der weltweite Markt von kleinteiliger Software, der durch das Internet zugänglich ist, Bestandteil des lokalen Austauschs von Konfigurationen (Mackay, 1990). Provisioning 2.0 sollte diesen Entwicklungen Rechnung tragen und agile Projektteams dabei unterstützen, die weltweite User- und Producer-Community in die selbst-organisierte Arbeitsplatzgestaltung zu integrieren.

Der vorgestellte Prototyp Peerclipse unterstützt diese Entwicklung indem er das Aneignen neuer Plug-ins durch die Innovatoren im Team transparenter macht und das (Ver-)Teilen im Team vereinfacht. Durch den Aufbau virtueller Gruppen-Repositories wird zudem die Verständigung über Gruppenbedarfe vereinfacht und das Bewusstsein für lokal vorhandene Werkzeug-Expertise gesteigert. Im Weiteren wollen wir Peerclipse im größeren Kontext erproben und die eingesetzten Recommender Algorithmen verfeinern. Dabei gilt es insbesondere lokale und globale Netzwerke der Anwender-Community Eclipse besser miteinander zu verzahnen.

Literatur

Bolte, A. und S. Porschen (2006). Die Organisation des Informellen, VS Verlag.

Buchsein, R., F. Victor, et al. (2008). IT-Management mit ITIL® V3. Wiesbaden, Vieweg + Teubner.

Budweg, S., G. Stevens, et al. (2008). "Medium und Mechanism" - Zur Rolle von Koordinatoren in der Praxis. Konferenz Mensch & Computer (M&C).

Coupaye, T. und J. Estublier (2000). Foundations of Enterprise Software Deployment. Proc. of CSMR, IEEE Computer Society.

Denzin, N. K. (1978). The research act: A theoretical introduction to sociological methods. New York, Praeger.

Draxler, S., H. Sander, et al. (2008a). Plug-in recommending for Eclipse users. 18th ECAI - Workshop on Recommender Systems, Patras.

EDC (2008). Users' Choice IDEs - 2008: A comprehensive study over 1200 software developers. Santa Cruz, Evans Data Corporation.

Gantt, M. und B. A. Nardi (1992). Gardeners and gurus: patterns of cooperation among CAD users. Proc. of CHI'92.

George, J. F., S. Iacono, et al. (1995). "Learning in Context: Extensively Computerized Work Groups as Communities-of-Practice." Accounting, Management and Information Technology 5(3-4): 185-202.

Gerlach, D., N. Güven, et al. (2007). Vergleich von Provisioning-Tools, Universität Stuttgart.

Kahler, H. (2001). "More Than WORDs - Collaborative Tailoring of a Word Processor." J. UCS 7(8): 826-847.

Kahler, H. (2001). Supporting collaborative tailoring. Roskilde, Roskilde University, Denmark.

Mackay, W. (1990). Users and customizable Software: A Co-Adaptive Phenomenon. Boston (MA), MIT.

MacLean, A., K. Carter, et al. (1990). User-tailorable systems: pressing the issues with buttons. Proc. of CHI, 175-182.

Pipek, V., G. Stevens, et al. (2008). Towards an Appropriation Infrastructure: Supporting user creativity in IT adoption. ECIS 08, Galway, Ireland.

Reiswich, E. (2008). Remote Management von RCP-Anwendungen. Universität Hamburg. Diplomarbeit.

Rogers, E. (2003). Diffusion of Innovations. New York, Free Press.

Schwartz, T. (2007). Praxisgerechte Unterstützung kooperativer Aneignung am Beispiel der Eclipse IDE. Siegen, Universität Siegen.

Stevens, G. and S. Draxler (2009). Appropriation of the Eclipse Ecosystem: Local Integration of Global Network Production. COOP. Aix-en-Provence, Springer.

Twidale, M. B. (1999). Over-The-Shoulder Learning: supporting brief informal learning embedded in the work context Technical Report ISRN UIUCLIS.

Wolf, U. (2003). "Fünf Softwaremanagement-Lösungen im Kurzporträt: Verteiler-Zentrum." Linux-Magazin 07.

Won, M. und V. Wulf (2003). "Anpassungsumgebung für komponentenbasierte Software: Kooperativ und lernförderlich." icom 1: 28-34.

Wulf, V. (1997). Konfliktmanagement bei Groupware. Braunschweig Vieweg.

Wulf, V. (1999). "Let's see your Search-Tool!" - Collaborative use of Tailored Artifacts in Groupware. Proceedings of GROUP '99.