Aneignungspraktiken von Software-Entwicklern beim Offshoring

Fallstudie eines kleinen deutschen Softwareunternehmens

Alexander Boden, Sebastian Draxler, Volker Wulf

Institut für Wirtschaftsinformatik und Neue Medien, Universität Siegen

1 Einleitung

Die Auslagerung von Teilen der Softwareentwicklung im Rahmen von Software-Offshoring ist eine zunehmend verbreitete Geschäftsstrategie in der deutschen IT-Branche. Dabei zählen Kostenersparnis, aber auch Zugriff auf qualifiziertes Fachpersonal sowie die Umsetzung von Umstrukturierungen durch eine Fokussierung auf Kernkompetenzen zu den wichtigsten Motiven für deutsche Unternehmen. Gleichzeitig sind jedoch auch Risiken bekannt, die das Arbeiten in international verteilten Entwicklungsteams mit sich bringt: neben dem Einfluss geographischer Distanz und verschiedenen Zeitzonen zählen dazu insbesondere soziokulturelle, rechtliche sowie praxisbezogene Unterschiede zwischen den Standorten (King & Torkzadeth 2008).

Für das Management dieser Risikofaktoren setzen Unternehmen häufig in erster Linie auf organisatorische und technische Maßnahmen, etwa Prozessoptimierungen, den Einsatz von Kooperationstools zur Planung der Arbeitsteilung, Überwachung des Projektfortschritts, etc. Neben diesen stark strukturierenden Werkzeugen werden Kommunikationsmedien wie E-Mail und insbesondere Instant Messenger Tools für ad hoc Koordination eingesetzt. Während die Anwendung sowie das Design entsprechender Werkzeuge bereits Gegenstand zahlreicher Studien sind, beschäftigen sich wenige Forschungsarbeiten mit dem Problem der Diffusion und Aneignung neuer Softwarewerkzeuge durch die, in diesem Fall verteilten, Projektmitarbeiter.

Um Aneignungspraktiken in verteilt arbeitenden Teams besser zu verstehen, haben wir eine Fallstudie in einem kleinen deutschen Softwareunternehmen durchgeführt das eng mit einem Partnerunternehmen in Tomsk, Sibirien kooperiert. Insbesondere gehen wir den Fragen nach, wie sich Entwickler in kleinen Softwareunternehmen neue Werkzeuge und Technologien aneignen, wie entsprechendes Wissen zwischen den Mitarbeitern ausgetauscht wird und in welchem Wechselspiel

die firmenspezifischen Praktiken des Unternehmens mit den Rahmenbedingungen verteilter Zusammenarbeit stehen.

Das Papier ist wie folgt aufgebaut: Abschnitt 2 gibt einen Überblick über relevante Literatur, insbesondere zum Konzept der Artikulationsarbeit, sowie zur Aneignung von Softwarewerkzeugen. Abschnitt 3 erläutert unsere methodische Vorgehensweise. Abschnitt 4 stellt die Fallstudie sowie die von uns im Feld identifizierten Praktiken dar, die anschließend in Abschnitt 5 analysiert werden. Das Papier endet mit einer Zusammenfassung der Ergebnisse sowie Designempfehlungen für die Unterstützung der positiven Effekte von Aneignung in Abschnitt 6.

2 Literaturanalyse

2.1 Kooperation und Kooperationssysteme beim Software-Offshoring

Offshore-Outsourcing (kurz: Offshoring) von Teilen der Softwareentwicklung ist in den letzten Jahren für eine wachsende Zahl kleiner und mittelständischer Unternehmen der deutschen IT-Branche zum Thema geworden (Dibbern & Heinzl 2006). Aufgrund ihrer geringen Größe und der oft auf starke Kundenbindung ausgerichteten Geschäftsmodelle, setzen KMU häufig auf sehr flexible und informelle Formen von Koordination und Kooperation. Gerade Softwareentwicklung benötigt dabei ein hohes Maß an Reaktionsvermögen, um unerwarteten Entwicklungen dynamisch begegnen zu können.

Um sich nicht zu stark festlegen zu müssen, verzichten KMU beim Offshoring häufig bewusst auf die Etablierung formalisierter Prozessreifegrade, die als zu kostenintensiv und zu inflexibel empfunden werden (BMBF 2000). Die Alternative, der Einsatz agiler Entwicklungsmethoden, ist jedoch erheblich von intensiver Kommunikation und Team-Interaktion abhängig und benötigt von daher spezielle Toolunterstützung, um den Herausforderungen international verteilter Teamarbeit begegnen zu können (Ågerfalk & Fitzgerald 2006; Herbsleb et al. 2005). Dabei zeigten bisherige Studien, dass KMU oftmals auf den Einsatz spezialisierter Kooperationssysteme weitgehend verzichten. Stattdessen setzen Unternehmen für die Koordination ihrer verteilten Entwicklungsprojekte häufig auf gewachsene Kombinationen unterschiedlicher Werkzeuge wie Instant Messenger Tools, Defect-Tracking Systeme und einfache Textverarbeitungsprogramme (Boden et al. 2007). Die damit verbundenen Arbeitspraktiken sind eng in informelle ad hoc Koordinationsprozesse eingebunden, die als *Articulation Work* (Artikulationsarbeit) im Sinne von Anselm Strauss verstanden werden können (Boden et al. 2009).

Das Konzept der Artikulationsarbeit stellt einen breiten Ansatz zur Beschreibung von Koordinationsleistungen dar, die sowohl auf formaler, als auch unterhalb der formal geregelten Ebene geleistet werden. Der Fokus von Artikulationsarbeit liegt dabei insbesondere auf der Verteilung und Zusammenführung von Teilaufgaben: wer macht was, bis wann, in welcher Qualität, unter Einsatz welcher Ressour-

cen etc. (Strauss 1988). Im Gegensatz zum klassischen Verständnis von Koordination, die häufig weisungsorientiert, hierarchisch und formal verstanden wird, umfasst Artikulationsarbeit dabei jedoch auch situierte Abstimmungsprozesse auf der Mitarbeiterebene (Schmidt & Simone 1996). Kurz gesagt, umfasst Artikulationsarbeit all die notwendige Arbeit, die für eine erfolgreiche Kooperation geleistet werden muss – also nicht nur die Verteilung von Aufgaben, sondern auch die Ermöglichung von deren Durchführung, sowie die Wiederzusammenführung von Teilergebnissen. Zur Artikulationsarbeit zählen daher nicht nur konkrete Abstimmungsprozesse, sondern der gesamte Metabereich kooperativer Arbeitskonstellationen. Also beispielsweise auch die Installation und Aneignung von Kooperationstools sowie entsprechender Technologien und Infrastrukturen, die in diesem Paper fokussiert werden sollen.

2.2 Aneignung von Software

Das Verhältnis zwischen Entwicklung und Nutzung von Software-Artefakten, sowie die damit verbundenen Auswirkungen auf das soziotechnische Umfeld sind immer wieder Thema der HCI und CSCW Forschung gewesen. Wir beziehen uns hier insbesondere auf die Aneignungsforschung und den Aneignungsbegriff nach Pipek (2005). Aneignung steht dabei in Relation zu Konzepten aus dem Bereich der Forschung zu Anpassbarkeit (*Tailorability*), aber auch des Bereichs der Einführung von Software in Organisationen in Bezug auf die dabei relevanten Veränderungen der Arbeitspraxis von Organisationen.

Der Begriff der Tailorability entstammt der Designorientierten Forschung und beschreibt ein Konzept um die Nutzer in den Prozess der Gestaltung von Software einzubeziehen. Während andere Methoden der klassischen Softwareentwicklung dies durch z. B. Workshops mit Nutzern erreichen, versucht Tailorability die Mitgestaltung durch Nutzer bis in die Nutzungsphase hinein zu verlängern. Nutzer sollen durch Anpassungsmöglichkeiten in die Lage versetzt werden, Software nach ihrer Auslieferung selbst an ihre persönlichen Bedürfnisse anzupassen. Klassische Beispiele reichen heute von der Veränderung der Schriftgröße im Web Browser, über das Anpassen von Menüleisten in Office-Anwendungen, bis zum nachträglichen Installieren und Einrichten von zusätzlichen Plug-ins in Firefox oder Eclipse. Das Verhältnis von Nutzung und Anpassung ist dabei häufig problematisch, da nicht immer scharf zwischen Benutzung und Anpassung getrennt werden kann.

Eine besondere Rolle spielt dabei die kooperative Anpassbarkeit. Diese umfasst die Möglichkeit, Anpassungen wie Konfigurationsdateien oder Wissen mit anderen Mitarbeitern zu teilen. Diese Technik hatte zu Beginn noch einen technischen Fokus. Sie wurde als Möglichkeit verstanden, die neue Hürde der Komplexität zu reduzieren, welche Anpassbarkeit mit sich bringt (MacLean et al. 1990). Spätere Arbeiten widmeten sich demgegenüber eher der Einbettung kooperativer Anpassbarkeit in Organisationen und den dabei auftretenden sozialen und soziotechnischen Effekten (Mackay 1990; Gantt & Nardi 1992). Mackay, wie auch Gantt

und Nardi, weisen dazu den Mitgliedern von Organisationen bestimmte Charakterisierungen zu, beispielsweise *Innovators*, *Early Adopters* oder *End Users*. Dabei wird z. B. Innovators eine hohe Affinität für das Ausprobieren neuer Software zugeschrieben, während *Early Adopters* zwar nicht so affin sind, aber als Multiplikatoren für Endbenutzer fungieren. Insbesondere die Studie von Orlikowski und Hofman (1997) zeigte, dass die Auswirkungen der Einführung von Software (Anpassungen möchten wir hier hinzuzählen) neben den geplanten Veränderungen auch ungeplante, durch die Nutzer hervorgerufene Veränderungen umfassen. Bei richtigem Management von Veränderungen können diese Effekte unter Umständen zum Vorteil genutzt werden.

Pipeks Verständnis der Aneignung von Software ist eng mit diesen Begriffen und Erkenntnissen verknüpft. Einerseits beschreibt er es als ein (ganzheitlich gesehen) kooperatives Unterfangen: Aneignung wird hier als *"assignment of purpose or use"* verstanden. Es handelt sich dabei also um eine Deutung oder Umdeutung der intendierten Nutzung, bzw. eine Weiterentwicklung der Nutzung, einschließlich der dafür notwendigen Maßnahmen, wie etwa Tailoring. Pipek nennt aber auch eine zweite Interpretation von Aneignung, die er als genauso wichtig ansieht: *"to take or make use of (something) without authority or right"*. Aneignung von Software kann also über das geforderte oder erwünschte Maß an Deutung, Umdeutung oder Diffusion neuer Nutzungsformen hinaus gehen (Pipek 2005). Für den Bereich der Softwareentwicklung hat das Konzept der Aneignung einen besonderen Stellenwert, da sich die Nutzer hier fast ausschließlich auf Software Werkzeuge verlassen. Dabei stellt sich die Frage, wie sich die Verteiltheit der Teams beim Offshoring auf die Aneignungspraktiken der Softwareentwickler auswirken.

3 Methodik

Zur Vorbereitung unserer Untersuchung wurde eine Analyse der vorhandenen Literatur zum Offshoring von Softwareentwicklung durchgeführt. In Bezug auf KMU wurden dabei Forschungsfragen identifiziert, die auf Aspekte der Artikulationsarbeit in Form informeller Koordinations- und Kommunikationspraktiken, sowie auf Aspekte der Aneignung von Software in KMU zielten. Auf dieser Grundlage wurde eine Interviewerhebung in zehn deutschen KMU durchgeführt, mit dem Ziel, Praktiken und Strategien kleiner Unternehmen mit Offshore-Softwareentwicklung identifizieren und einordnen zu können. Gleichzeitig wurde aus dem Sample ein Unternehmen als besonders interessanter Fall identifiziert und im Rahmen der weiteren Studien eingehender untersucht.

Da Artikulationsarbeit nur aus der Praxisperspektive heraus vollständig verstanden werden kann, bildeten teilnehmende Beobachtungen den Schwerpunkt der weiteren Untersuchung. Dazu wurde das deutsche Unternehmen über einen Zeitraum von zwanzig Arbeitstagen besucht, das russische Partnerunternehmen für den Zeitraum einer Woche. Vor Ort konnten auf diese Weise lokale und verteilte

Artikulationspraktiken im Rahmen von Meetings, individuellen Arbeitssituationen und kooperativen Aufgaben *in situ* beobachtet werden. Ergänzend zur Beobachtung wurden ausführliche Interviews mit dem Inhaber des Unternehmens sowie mit Entwicklern und Projektleitern durchgeführt, außerdem Artefakte wie E-Mails, Chat-Protokolle, interne Arbeitspapiere und Tafelbilder ausgewertet.

Die Ergebnisse der Untersuchung wurden in Form von Feldnotizen und Fotos dokumentiert, die während der teilnehmenden Beobachtung angefertigt wurden. Zur Auswertung des Datenmaterials orientierten wir uns am *Grounded Theory*-Verfahren nach Glaser und Strauss (1996). Dazu wurde das erhobene Material jeweils im Anschluss an die einzelnen Interviews und Teilnehmenden Beobachtungen ausgewertet und entsprechend des Fokus unserer Untersuchung kodiert. Dabei wurden zunächst auf der Grundlage der Ergebnisse Kategorien gebildet. Diese wurden anschließend miteinander in Beziehung gesetzt und im Rahmen der Analyse iterativ weiterentwickelt.

4 Ergebnisse

4.1 Die Fallstudie „Alpha“

Das von uns beforschte KMU ist ein Dienstleister im Bereich Datenverarbeitung, Statistik und Dokumentation. Es kann auf eine mehr als zehnjährige Offshoringkooperation zurückblicken. Das Unternehmen beschäftigt etwa 20 Mitarbeiter. Produkte umfassen Datenbanken sowie Erschließungs- und Dokumentationssysteme, die unter anderem in Kultureinrichtungen wie Archiven und Museen eingesetzt werden. Alpha legt Wert auf flache Hierarchien und flexible, selbstverantwortliche Arbeit. Die Projekte laufen in der Regel in speziell dafür abgestellten Projektteams und in Form von längeren Kooperationen in enger Symbiose mit den Projektleitern des Kunden. Dabei passt sich das Unternehmen hinsichtlich der Entwicklungsmodelle den Vorgaben und Wünschen des Kunden an, legt selbst aber keinen Wert auf deren Einsatz.

Seit Ende der 1990er Jahre hat das Unternehmen durchschnittlich vier bis acht Softwareentwickler in Tomsk, Sibirien beschäftigt. Grundlage für die Entscheidung, einen russischen Partner anzuwerben, war ein Praktikum eines fähigen russischen Entwicklers im Unternehmen. Auf Grundlage der positiven Erfahrung mit dem Praktikanten entschloss man sich, Offshoring zu betreiben. Konkret ging es dabei zunächst um das *Reengineering* einer bestehenden Software, später wurden zusätzliche Projekte hinzugenommen.

Für die Kooperation zwischen den Teams stellt das Unternehmen mehrere Softwarewerkzeuge bereit, zu denen, neben einem eigenen Email- und Kalenderserver, insbesondere das Projektplanungstool *dotProject*, das Defect-tracking System *Mantis*, sowie das Versionsverwaltungssystem *Subversion* zählen. Dabei macht der Chef des Unternehmens jedoch (abgesehen von der geforderten Nachhaltung der

Zeitaufwände per dotProject) wenig Vorgaben für die Verwendung der Tools, vielmehr überlässt er den einzelnen Projektleitern die Entscheidung darüber, welche Tools sie wie für die Abwicklung ihrer Projekte verwenden sollen (jedenfalls solange diese keine teuren Lizenzen erfordern).

4.2 Strategische Entscheidungen für neue Softwarewerkzeuge

Im Rahmen unserer Studie konnten wir mehrere Fälle strategischer Entscheidungen für neue Softwarewerkzeuge beobachten. Einen besonders interessanten Fall stellte dabei ein vom deutschen Management angesetztes Strategiemeeting im Jahr 2006 dar, bei dem die deutschen Entwickler gemeinsam mit einigen extra zu diesem Zweck angereisten russischen Mitarbeitern über die Möglichkeit diskutieren sollten, zukünftige Versionen eines gerade fertig gestellten Softwareproduktes mit *Eclipse* in Java zu realisieren. Um die Entscheidung vorzubereiten, hielt einer der deutschen Entwickler, der im Rahmen eines kleinen anderen Projektes bereits mit Eclipse gearbeitet hatte, ein eineinhalbtägiges Tutorial für die deutschen und russischen Mitarbeiter ab, wobei er auf Fragen des Aufbaus des Programms, sowie dessen Funktionen und die Bedienung einging. Er erläuterte dazu: *„letzten Endes (war) der Auftrag zu zeigen, was ist technisch möglich, was ist machbar, was ist im Augenblick der Stand der Dinge, was machen andere, und auch die Möglichkeiten aufzeigen, die man mit so einer Vorgehensweise hat."*

Während des Meetings sollten sich also die anwesenden deutschen und russischen Entwickler ein Bild von der Funktionsweise und den Möglichkeiten von Eclipse machen können, um diese vor dem Hintergrund ihrer bisherigen Entwicklungserfahrungen, sowie ihrer Antizipation über das zukünftige Produkt zu reflektieren. Dabei diskutierten die Entwickler die Vor- und Nachteile von Eclipse und Java, wobei insbesondere die komfortablere Bedienung, Unterstützung durch "Wizards", sowie die unter Java verbesserte „inkrementelle Kompilierung" und „hot code replacement" als wichtigste Vorteile für die alltägliche Entwicklungsarbeit hervorgehoben wurden. Neben diesen auf die alltägliche Entwicklungsarbeit bezogenen Erwägungen, spielten bei dem Treffen zusätzlich auch strategische Überlegungen des Unternehmens eine Rolle: so bietet Eclipse als Rich Client Plattform die Möglichkeit, Plugins als selbstständig lauffähige Produkte weitgehend plattformunabhängig zu realisieren. Außerdem wurde hervorgehoben, dass die Eclipse zugrundeliegende Komponenten-Architektur sich sehr gut dafür eignet, Komponenten für weitere Projekte wieder zu verwenden. Dies ist sinnvoll, da spezielle Kundenwünsche meistens individuelle Softwarelösungen erfordern, die häufig wiederkehrende Features und Funktionen aufweisen, und somit effizient aus Einzel- Komponenten zusammengestellt werden können.

Ähnliche Aneignungsprozesse konnten wir auch in weiteren Fällen beobachten, wo russische Entwickler vor Ort in Deutschland mit ihren Projektleitern eine einheitliche Entwicklungsplattform für neue Projekte einrichteten. Im konkreten Fall sollte das Unternehmen ein auf *Apache* und *Tomcat* basierendes Internetportal

für die Präsentation von Archivmaterial entwickeln und dabei eine auf *Lucene* basierende Suchfunktion bereitstellen. Projektleiter war ein deutscher Mitarbeiter, der mit einem erst kürzlich eingestellten russischen Entwickler gemeinsam das Projekt abwickeln sollte. Dieser hatte sich zum Zeitpunkt unserer Beobachtung bereits in das Programm eingearbeitet und hatte aktuelle Versionen der nötigen Serverinfrastruktur, sowie einige aus einem anderen Projekt vorhandene Komponenten auf seinem Notebook installiert.

Der Besuch in Deutschland sollte nun dazu dienen, eine identische Installation auf dem PC des deutschen Projektleiters einzurichten, mit dem Ziel, eine gemeinsame Arbeitsplattform für die (verteilte) Entwicklungsarbeit zu generieren. Dabei übernahm in erster Linie der russische Entwickler die Installation der unterschiedlichen Tools auf dem Rechner des deutschen Projektleiters. Gleichzeitig erläuterte er seine Dateiablage- und Entwicklungsstrategie, wozu er den deutschen Projektleiter auch auf verschiedene Tools zum Verwalten und Vergleichen von Dateien hinwies. Für die Einrichtung wurden diese auch auf dem Rechner des Projektleiters installiert, und der russische Entwickler erläuterte deren Gebrauch.

Dabei wurde deutlich, dass im Rahmen dieser Treffen auch ein Techniktransfer zwischen den Teams stattfand, indem deutsche Entwickler den russischen Kollegen neue Technologien erläuterten (etwa im Rahmen des Strategiemeetings zu Eclipse), oder wie in diesem Fall, durch Empfehlungen russischer Entwickler. So wurde uns auch davon berichtet, dass der Einsatz von Instant Messenger Tools im Unternehmen stark auf der Initiative der russischen Mitarbeiter beruht, die kurz nach Beginn der Kooperation das Tool den deutschen Kollegen vorgestellt und immer wieder für dessen Einsatz plädiert hatten. Mittlerweile haben die meisten deutschen Entwickler das Tool installiert und schätzen es sehr als nützliches Hilfsmittel für informelle, wenig intrusive Nachfragen, das Übermitteln von Statusinformationen und Links, sowie als Nachschlagewerk für geführte Unterhaltungen (durch die automatische Protokollierung der Gespräche).

4.3 Ad hoc Aneignungspraktiken von neuen Softwarewerkzeugen

Bei der Beobachtung von geplanten Arbeitstreffen zwischen deutschen und russischen Mitarbeitern (wie in 4.2 beschrieben) zeigte sich deutlich, dass neben formaler Artikulationsarbeit auch informelle Abstimmungsprozesse bezüglich der Nutzung von Werkzeugen und Technologien eine Rolle spielten, die nicht im intendierten Fokus der Arbeitstreffen standen. Ähnliche, sehr situativ und wenig strukturiert ablaufende Artikulationsprozesse, konnten wir im Rahmen unserer Studie mehrfach beobachten und thematisierten dies auch in unseren Interviews. Dabei zeigte sich, dass die entsprechenden Lernprozesse stark auf bestimmten Mitarbeitern beruhen, die Experten in bestimmten Bereichen sind und andere Entwickler anlernen.

Der Rahmen für entsprechende Ereignisse waren die regelmäßigen Meetings, wie der wöchentlich veranstaltete *Jour-fixe*, bei dem die einzelnen Mitarbeiter das Team

über aktuelle Entwicklungen auf dem Laufenden halten sollten. Dabei wurden auch immer wieder neue Technologien und Softwarewerkzeuge vorgestellt, welche nützlich für die Arbeit der gesamten Gruppe sein könnten. Interessant war dabei, dass die Mitarbeiter in den Interviews die eigentlichen Treffen nicht als sehr nützlich empfanden, da die dort vermittelten Informationen häufig entweder zu allgemein, oder zu detailliert sind. Zudem ist der Fokus sehr uneinheitlich: *„Viele Kollegen breiten das dann aus mit, ich hab mit dem telefoniert, und mit demjenigen. Das find ich jetzt persönlich nicht so interessant. (...) Und häufig geh ich einfach ein, zweimal am Tag rum, und frag. Also, meistens ist es so, dass ich einfach frage wo stehst du, und dann, was macht ihr gerade so ...“* Die auf den Treffen vermittelten Informationen werden dabei mitunter als Anknüpfungspunkte für spätere Gespräche genutzt, bei denen dann auch der Einsatz von Softwarewerkzeugen thematisiert werden kann. So berichtete einer der deutschen Entwickler im Rahmen eines Interviews: *„Mhm, also dadurch dass ich viel rumlaufe, und auch einfach frage, was machst Du gerade. Dann heißt es, ja ich schreib ne Anwendung, ich muss Screenshots machen. Dann sag ich, ja hier wunderbar, mach mal damit, dann ist die Arbeit einfacher, oder auch nicht. Und dann setzt's sich durch, oder halt auch nicht (lacht).“*

Der dabei stattfindende Wissensaustausch ist stark situativ geprägt und zudem vom persönlichen Einsatz bestimmter Mitarbeiter abhängig, die auf eigene Initiative im Unternehmen umhergehen, beobachten, und Tipps geben. Gleichzeitig entwickeln diese Mitarbeiter jedoch auch einen besonderen Status im Unternehmen und werden von Kollegen im Fall von Problemen häufiger als andere angesprochen.

Entsprechender Wissensaustausch in Bezug auf Tools kann aber auch indirekt im Rahmen von eigentlich auf die Entwicklungsarbeit bezogener Artikulationsarbeit stattfinden. Diese findet häufig informell statt, da die im Unternehmen verwendete Technik wenig dokumentiert ist. Darüber hinaus sind die vorhandenen Informationen stark fragmentiert und häufig schwer zu finden. Aus diesen Gründen wenden sich die Mitarbeiter im Fall von Problemen lieber direkt an einen als kompetent geltenden Kollegen, statt Informationen in Datenbanken, E-Mails oder Chat-Logs zu suchen. Dieses Vorgehen schafft immer wieder Gelegenheiten für den Austausch, auch über Softwarewerkzeuge und Technologien im Unternehmen. In Bezug auf die geographische Distanz zwischen den Teams kommt es dabei jedoch auch immer wieder zu Problemen, beispielsweise wenn ein Team dringend eine Information von Seiten des entfernten Entwicklungsstandortes benötigt und keine Antwort erhält, was laut einigen Entwicklern von Unternehmen Alpha keine Seltenheit darstellt.

Um diese Probleme verteilter Zusammenarbeit zu adressieren, pflegt das Unternehmen einen regelmäßigen Austausch von Entwicklern zwischen Russland und Deutschland. So werden neu eingestellte russische Mitarbeiter mindestens einmal für einen Zeitraum von einigen Wochen nach Deutschland eingeladen, um das persönliche Kennenlernen zu fördern. Zudem reist der deutsche Chef sowie einige der Projektleiter regelmäßig nach Russland, insbesondere zu Beginn neuer Projekte

oder kurz vor deren Abschluss. Zudem bietet das Unternehmen besonders fähigen russischen Entwicklern die Möglichkeit, den deutschen Standort auch über einen längeren Zeitraum zu besuchen um dort das Team zu verstärken. Dabei sollen die Aufenthalte unter anderem auch den Wissensaustausch fördern, indem russische Entwickler als lokale Ansprechpartner vor Ort fungieren können. So berichtete einer der russischen Mitarbeiter in Deutschland über den Zweck seines Aufenthaltes: *„As I told you, he (the German project leader) is (...) responsible for the design of (product X). But (product X) is based on the framework that was created by me, and partly by other colleagues in Tomsk. So (...) I also work as an expert for him. He asks me, I answer. And, also I stimulate him to invent better technical solutions in this project."* Durch den Aufenthalt des russischen Entwicklers in Deutschland entsteht also die Möglichkeit für informelle Nachfragen zu ad hoc auftauchenden Fragen bezüglich der Entwicklungsarbeit, welche andernfalls wesentlich schwieriger zu handhaben wären.

5 Diskussion

Im Rahmen der vorgestellten Studie nutzten wir Strauss' Konzept der Articulation Work als Analytische Linse, um Aneignungspraktiken von Software empirisch nachzuweisen. Dabei konnten wir zwei unterschiedliche Strategien der Aneignung und Diffusion von Software Artefakten und Nutzungspraktiken innerhalb der Organisation beobachten.

Einerseits wurden geplante Änderungen, wie der Umstieg auf die Eclipse Plattform, im Rahmen formaler Meetings von den Beteiligten diskutiert. Andererseits stießen wir jedoch auch auf ungeplante, ad hoc auftretende Aneignungsprozesse, wie beispielsweise die Aneignung neuer Tools und Funktionalitäten im informellen Rahmen der alltäglichen Zusammenarbeit. Dies war der Fall bei der Einrichtung einer gemeinsamen Entwicklungsumgebung für ein konkretes Projekt. Unsere Studie zeigte dabei, dass Aneignungspraktiken im Rahmen informeller Artikulationsarbeit eine wichtige Rolle spielen können, da auf diese Weise z. B. *Grassroots*-Innovationen für die Kooperation nutzbar gemacht werden können. Ein Beispiel dafür ist etwa die (völlig ungeplante) Diffusion von Instant Messenger Tools im Unternehmen, die für die Koordination der Teams beim Offshoring eine sehr wichtige Rolle spielen (vgl. Boden et al. 2007).

Unsere Beobachtungen decken sich dabei grundsätzlich mit existierenden Erkenntnissen, z. B. (Orlikowski & Hofman 1997), dass die Einführung von Software nicht komplett formal planbar ist. Vielmehr sollten Unternehmen bei der Aneignung neuer und vorhandener Werkzeuge ein ausgewogenes Verhältnis zwischen formal gewünschten Änderungen und informell auftretenden und emergenten Aneignungsvorgängen angestrebt werden. Die praktische Ausgestaltung dieser Vorgänge ist dabei stark von der Art und Weise abhängig, wie Unternehmen ihre Projektsteuerung (d.h. die notwendige Artikulationsarbeit) organisieren, und wie viel Freiräume den Entwicklern eingeräumt werden hinsichtlich der Abwicklung

ihrer Arbeit. Gleichzeitig wurde deutlich, dass bestimmte Mitarbeiter als *Early Adopters* (vgl. Mackay (1990) und Gantt u.a. (1992)) eine zentrale Rolle einnehmen, für die (oft situative) ad hoc-Aneignung von neuen Tools sowie die Einarbeitung neuer Mitarbeiter.

Die Aneignung von Tools mittels formaler und geplanter Maßnahmen funktionierte auch über Teamgrenzen hinweg ausreichend. Entsprechende Prozesse wurden dabei im Rahmen von persönlichen Treffen (beider Unternehmensteile) an einem der Standorte angeregt. Allerdings können in der Regel nur bestimmte Personen (oftmals Projektleiter statt *Early Adopters*) das andere Team besuchen. Dabei sind persönliche Treffen nur selten möglich und finden häufig unter starkem Zeit- und Arbeitsdruck statt (z. B. kurz vor Projektabschluss, wenn dringend Ergebnisse benötigt werden). Obwohl Mitarbeiter beider Teams berichteten, dass diese Treffen sehr wichtig und lehrreich sind, bieten sie zudem häufig allenfalls einen Einstieg oder Überblick in neue Technologien. Die für die eigentliche Aneignung von Software und Nutzungspraktiken wichtigen informellen und ungeplanten Austauschprozesse, die mit der Einführung neuer Software einhergehen, konnten dagegen in diesem Fall in erster Linie lokal am Arbeitsplatz oder in der Kaffeeküche beobachtet werden. Zwischen den Teams dagegen findet dieser Austausch weitaus weniger statt, und mitunter werden sogar direkte Anfragen zu spät oder gar nicht beantwortet.

Die Rolle der informellen Aneignung im Rahmen von situativer Artikulationsarbeit wird demgegenüber oft übersehen, sollte aber für eine Durchdringung des Unternehmens mit den eingesetzten Technologien gefördert werden. Pipek (2005) sieht hier zwei grundsätzliche Möglichkeiten Aneignung zu unterstützen: Einerseits die Einführung einer mediierenden Rolle, die doppelt qualifiziert, sich mit Technik und der Arbeitspraxis auseinandersetzt und andererseits die Stimulation des Austausches von Software und Nutzungsgewohnheiten.

Das von uns untersuchte Unternehmen versuchte den Austausch zwischen den Standorten durch den längerfristigen Austausch von Projektmitarbeitern zu unterstützen, die man als *Early Adopters*, als zentrale Personen für Innovation im Unternehmen einstufen könnte. Diese Strategie zeigt durchaus Ähnlichkeiten zum Konzept der *Knowledge Broker* oder Wissensvermittler, die im Bereich des Wissensmanagements beschrieben wurden und die Vermittlerfunktionen zwischen den Teams oder sozialen Netzwerken einnehmen (Milewski et al. 2008). Das hineinwachsen in solche Rollen geschieht häufig aus persönlichem Interesse, weshalb sich diese in der Regel aus der Praxis herausbilden, und nur schwer formal delegiert werden können. Meist sind es diejenigen Mitarbeiter, die bei Problemen häufiger angesprochen werden als Andere. Damit diese Rolle effektiv ausgefüllt werden kann, könnten Unternehmen z. B. in beiden Teams Mediatoren auswählen. Diese können dann Veränderungen explizit an Mitarbeiter weiter geben.

Der Austausch/Diskurs über Nutzungsgewohnheiten oder eingesetzte Werkzeuge kann dabei z. B. durch leichtgewichtige, flexible Tools wie z. B. ein internes Blog zum Thema Toolnutzung unterstützt werden. Zudem ist es gerade im Bereich von

Eclipse z. B. leicht möglich, transparent zu machen, wer welche Komponenten einsetzt. Diese Softwaretechnische Unterstützung könnte in der Zukunft insbesondere in verteilten Projekten eine Gruppenwahrnehmung schaffen, die dem Gespräch in der Kaffeeküche oder dem Blick über die Schulter der Kollegen nahe kommt um letztlich informelle Aneignungsprozesse anzustoßen.

6 Fazit

Wir konnten zeigen, dass die Konzepte der kooperativen Aneignung aus der HCI und CSCW Forschung auch in diesem Fall Gültigkeit haben. Dabei haben wir die wichtige Rolle informeller Kommunikation als Kanal für Aneignungsaktivitäten an verschiedenen Beispielen beobachten können, und dies in Relation zu formaleren Formen der Aneignung und strategischen, geplanten Einführung von Software gestellt. Das ist gerade im Zusammenhang mit Offshoring ein interessantes Ergebnis, da hier ja oft eine starke formale Steuerung als Erfolgsrezept beschrieben worden ist. Gleichzeitig illustriert unsere Studie, welche Probleme bei der informellen Aneignung von Werkzeugen und Technologien in verteilten Teams auftauchen können.

Die Unterstützung auch von informellen Aneignungspraktiken beim Software-Offshoring ist ein wichtiges Thema, das bisher wenig Beachtung erfahren hat. Dabei werden die von uns fokussierten Aspekte in der Zukunft eher zunehmen, da es einen deutlichen Trend hin zu agilen Entwicklungsmethoden sowie zu modularen, flexibel konfigurierbaren Softwarewerkzeugen zu verzeichnen gibt. Gleichzeitig müssen die Akteure immer mehr Konfigurations- und Anpassungsaufgaben selbst bewältigen, was eine entsprechende Unterstützung erfordert. Dabei ist ein detailliertes Verständnis von Artikulationsarbeit in verteilten Teams erforderlich, wozu unsere Fallstudie zu Software-Offshoring in einem deutschen KMU einen Beitrag leisten möchte.

Literatur

Agerfalk, P.J. & Fitzgerald, B., 2006. Flexible and distributed software processes: old petunians in new bowls? Communications of the ACM, 49(10), 27-34.

BMBF, 2000. Analyse und Evaluation der Softwareentwicklung in Deutschland, Available at: http://www.iese.fgh.de/pdf_files/evasoft_abschlussbericht.pdf

Boden, A., Nett, B. & Wulf, V., 2007. Coordination Practices in Distributed Software Development of Small Enterprises. In International Conference on Global Software Engineering. pp. 235-246.

Boden, A., Nett, B. & Wulf, V., 2009. Offshoring in kleinen und mittleren Unternehmen der Softwareindustrie. HMD. Praxis der Wirtschaftsinformatik, 265, 92-100.

Dibbern, J. & Heinzl, A., 2006. Selective Outsourcing of Information Systems in Small and Medium Sized Enterprises. In R. A. Hirschheim et al., eds. Information Systems Outsourcing. Berlin, Heidelberg: Springer, pp. 57-81.

Gantt, M. & Nardi, B.A., 1992. Gardeners and gurus: patterns of cooperation among CAD users. In Conference on Human factors in computing systems, pp. 107-117.

Herbsleb, J.D., Paulish, D.J. & Bass, M., 2005. Global Software Development at Siemens: Experience from Nine Projects. In Conference on Software Engineering, pp. 524-533.

King, W.R. & Torkzadeth, G., 2008. Information Systems Offshoring: Research Status and Issues. MIS Quarterly, 32/2.

Mackay, W.E., 1990. Patterns of sharing customizable software. In Conference on Computer-supported cooperative work. Los Angeles, California, pp. 209-221.

MacLean, A. et al., 1990. User-tailorable systems: pressing the issues with buttons. In Conference Human Factors in Computing Systems, pp. 175-182.

Milewski, A.E. et al., 2008. Guidelines for Effective Bridging in Global Software Engineering. In Conference on Global Software Engineering, pp. 23-32.

Orlikowski, W. & Hofman, D., 1997. An Improvisational Model for Change Management: The Case of Groupware Technologies. Sloan management review, 38/2, 11-21.

Pipek, V., 2005. From tailoring to appropriation support: Negotiating groupware usage, Oulu: University of Oulu.

Schmidt, K. & Simone, C., 1996. Coordinaton Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design. CSCW, 5, pp. 155-200.

Strauss, A.L., 1996. Grounded Theory: Grundlagen Qualitativer Sozialforschung, Weinheim: Beltz, Psychologie-Verl.-Union.

Strauss, A.L., 1988. The Articulation of Project Work: An Organizational Process. The Sociological Quarterly, 29(2), 163-178.