Kommunikation zwischen Entwicklern, Benutzern und Management

(Link zum Artikel: http://www.it-republik.de/dotnet/artikel/3051)

Früher nannte man es Kaffeeküche

Text: Uta Kapp und Jean Pierre Berchez
Die Kommunikation in einem Team findet auf vielen Ebenen statt. Es gibt einen direkten Austausch von Person zu Person. Der Wissensaustausch findet über indirekte Kommunikationswege und Dokumentenaustausch statt. Eine dritte Ebene ist prozessabhängig, und Benachrichtigungen werden automatisch per E-Mail an Teammitglieder versandt.
Teil 1 Teil 2

Die Qualität des Informationsflusses ist entscheidend für den Erfolg eines Softwareprojekts. Wissen ist Macht. Nur Wissen, das mit anderen geteilt wird, ist auch nützlich. Collaboration-Plattformen helfen uns dabei, Wissen zu teilen und stehen so als Informationsquelle bereit. Hier werden Informationen zu Anforderungen, Terminen, Releases, Builds und Tests gespeichert. Sie sind Wissensmanagementtools speziell für die Softwareentwicklung und damit mehr als nur Application-Lifecycle-Management-Werkzeugbanken.

Das, was uns auch heute in der sachlichen Wissensgesellschaft immer noch begeistert, sind Geschichten. Wie zu Urzeiten die Steinzeitmenschen in Mythen ihr Wissen am Feuer sitzend weitergaben, so stehen wir heute in der Kaffeeküche und erzählen unsere fantastischen Geschichten. Die Erzählkunst verpackt Wissen in einen größeren Kontext und vermittelt Fakten mit einer emotionalen Verpackung. Das berührt uns und vermittelt uns Erlebnisse, die zu tiefen Einsichten führen. So bleibt die direkte Kommunikation zwischen Personen die wichtigste Art des Informationsaustausches. In Meetings werden der aktuelle Projektstand sowie die zu bewältigenden Aufgaben und Konflikte besprochen. Wenn ein Team in der glücklichen Lage ist, am gleichen Ort oder vielleicht sogar im gleichen Großraumbüro untergebracht zu sein, fließt die Informationen zwischen den Beteiligten ständig.

Heute arbeiten viele Softwareentwicklungsteams verteilt über viele Standorte, offshore oder sogar als virtuelles Team. In Open-Source-Softwareprojekten ist dies üblich. In diesem Fall findet die direkte Kommunikation mithilfe von Chat, E-Mail, Telefon oder Videokonferenz statt. Schon der Austausch mit den Benutzern und Kunden oder Auftraggebern muss in einem Softwareprojekt organisiert stattfinden, wenn beide in verschiedenen Büros arbeiten.

Eine zweite, indirekte Ebene der Kommunikation sorgt für die Speicherung und Vermittlung von Wissen. Hier werden Informationen von einer Person in eine Wissensbasis eingegeben und von vielen anderen gelesen und ergänzt. Die Qualität der Verständigung, die zwischen Benutzern, Testern, Managern und Entwicklerteam läuft, ist der Schlüssel zu den Beziehungen zwischen allen. Idealerweise schafft der Einsatz einer Collaboration-Plattform die Kanäle für diesen Informationsfluss. Um die gleiche Erlebnistiefe zu erreichen wie die Erzählungen in der Kaffeeküche, brauchen wir weiterhin die alten Geschichten. Reine Fakten ohne Kontext sind für uns Menschen, die mit Logik und Gefühl ausgestattet sind, nicht verständlich. So sollte jedes Teammitglied zu einem Geschichtenschreiber werden. Harte

Fakten, Termine und Ereignisse vermischen sich mit subjektiven Wahrnehmungen und intuitiven Entscheidungen.

Die zentralen Organisationseinheiten für diese Informationen sind Arbeitseinheiten (WorkItems), die in intelligenten Listen (Trackern) arrangiert werden. Beispiele sind spezialisierte Arbeitseinheiten für Anforderungs- und Änderungswünsche. Benutzer können ihr Fachwissen zum Beispiel in einem Story-Card-Tracker weitergeben. Tester melden Fehler in einer Fehlerdatenbank, auch Bug-Tracker genannt. Wenn die Benutzer Zugriff auf solch eine Plattform haben, dann können sie den aktuellen Stand der reparierten Fehler einsehen, ohne ständig nachfragen zu müssen. Das entlastet die Arbeitsbeziehungen und den Druck auf die Entwickler.

Für jedes Softwarevorhaben gibt es einen Auftraggeber. Er ist derjenige, der für die Kosten aufkommt und für seine Investition das bestmögliche Produkt möchte. Der Auftraggeber ist meistens nicht identisch mit dem Benutzer. Er muss die Prioritäten setzen und sollte immer über den aktuellen Stand des Projekts informiert sein. Idealerweise sollte er auch Zugriff auf die Collaboration-Plattform haben. Bei Projekten, die als Auftragsarbeit vergeben wurden oder als Outsourcing laufen, ist das meist eine heikle Angelegenheit. Hier beeinflussen Randbedingungen, wie Verträge und Verhandlungsergebnisse, das Maß an Offenheit, das ein Projektteam bereit ist einzugehen. Zugriffsrechte der Collaboration-Plattform alleine reichen nicht aus, um dies zu regeln. Hier kommen Taktik und Strategie der zusammenarbeitenden Firmen ins Spiel.

Wenn ein Klima der Offenheit und Transparenz vorhanden ist, dann kann eine integrierte Plattform dafür sorgen, dass ein vertrauensvolles Miteinander entsteht. So entsteht eine qualitativ hochwertige Informationsquelle. Das ist ein Lernprozess der beteiligten Menschen. Für Produkte, bei denen das Team offshore, nearshore oder verteilt arbeitet, ist dies besonders wertvoll. Auch wenn die an der Entwicklung Beteiligten durch Zeitzonen getrennt sind, können Daten zu jedem Zeitpunkt in das System eingegeben und zu einem anderen Zeitpunkt abgerufen werden. Mitarbeiter, die eine Weile fehlen, weil sie zum Beispiel im Urlaub sind, können so problemlos nachvollziehen, was in ihrer Abwesenheit geschehen ist.

In Softwareentwicklungsprojekten gibt es noch eine dritte Kommunikationsebene, eine ereignisgetriebene Ebene. Wenn ein bestimmtes Ereignis eintritt, werden Mitglieder des Entwicklungsteams, der Anwender oder Kunden informiert. Diese Ereignisse können an verschiedenen Stellen auf der Collaboration-Plattform automatisch starten. Mitglieder der Plattform bekommen dann eine Information per E-Mail oder bei der nächsten Anmeldung am System.

Beispiele:

- Ein Fehler wird in dem Bug-Tracker gemeldet. Hier wird der dafür zuständige Projektleiter informiert, um den Fehler einzuschätzen und die Reparatur an einen Entwickler weiterzuleiten.
- Ein Fehler wird im Bug-Tracker als repariert gemeldet. Der Anwender, der den Fehler gemeldet hat, wird informiert.
- Der Tester meldet den Fehler als getestet. Dann wird der Anwender, der den Fehler gemeldet hat, darüber informiert.

- Ein automatischer Test schlägt bei einem automatisierten Build fehl. Hier wird der zuständige Entwickler unterrichtet.
- Ein Build ist fehlgeschlagen. Hier funktioniert die Kommunikation hochautomatisiert. Ein Benutzer stellt seinen Code in das Sourcecodearchiv und wird informiert, wenn dieser an irgendeiner Stelle im Build zu Schwierigkeiten führt.
- Ein neues Release ist fertiggestellt. Alle Benutzer werden informiert.

Es gibt eine Reihe von integrierten Collaboration-Plattformen, die alle notwendigen Anwendungen zu einem Paket geschnürt haben. Einige Beispiele hierfür:

SharePoint mit Visual Studio Team System 2008 Team Foundation Server von Microsoft: SharePoint integriert Web-2.0-Tools, wie Wiki und Blogs, mit einem Dokumentenmanagement, das sich nahtlos mit Microsoft Office Tools integriert. Hiermit lässt sich der Team Foundation Server kombinieren, um so ein ausgereiftes Application Lifecycle Management zu betreiben. Für die Entwicklung in der .NET-Welt ist diese Kombination eine ideale Lösung für Teams, die mit Microsoft Visual Studio 2008 arbeiten.

JAZZ/Rational Team Concert von IBM: JAZZ integriert Tools aus der IBM-Rational-Produktpalette, aber auch Anwendungen von Drittherstellern, zu einer einheitlichen Oberfläche. Rational Team Concert ist das Tool, das hier den Collaboration-Part übernimmt. Hier werden Projekte, Teams, WorkItems und Arbeitsabläufe verwaltet. Für den Entwickler ist hier die perfekte Einbindung in Eclipse als IDE besonders interessant.

CollabNet TeamForge von CollabNet Corporation: CollabNet TeamForge, vormals unter CollabNet SourceForge Enterprise auf dem Markt, ist eine komplett integrierte und offene ALM-Plattform mit einer Toolsuite für verteilt tätige Teams. Ob die Teammitglieder webbasiert arbeiten oder in eine IDE eingebunden sind, ist dabei egal. Die Anwendungen teilen sich ein zentrales Repository. Laut Hersteller ist so ist ein Produktivitätszuwachs von bis zu 50 % möglich, und Lizenz-, Administrations- und Rechnerkosten sollen so um bis zu 80 % schrumpfen.

Polarion von Polarion GmbH: Polarion ALM ist eine integrierte, browser- und Wiki-basierte Platform, die alle wesentlichen Aspekte des Application Lifecycle Managements vom Requirements, Change, Test und Configuration Management abdeckt. Für die Entwickler stehen Integrationen in MS Visual Studio und in die Eclipse IDE zur Verfügung. Damit kann in beiden Welten, Java und .NET, entwickelt werden.

MKS Integrity von MKS: MKS Integrity ist eine kollaborative Entwicklerplattform mit integriertem Kollaborations- und Application Lifecycle Management für verteilt arbeitende Softwareentwicklerteams. Interessant an MKS Integrity ist insbesondere die extreme Anpassbarkeit an eigene Bedürfnisse und das Abbilden von Entwicklungsprozessen. Es liefert automatisch eine „Traceability“ die im reguliertem Umfeld wie z.B. Medizintechnik aber auch Automotive oder gar im Finace Bereich immer wichtiger wird.

Neben den oben genannten integrierten Collaboration-Plattformen gibt es natürlich eine Menge von weiteren technischen Hilfsmitteln zur Kollaboration, wie die bekannten Werkzeuge der Firma Atlassian (JIRA, Confluence etc.) oder Tools, die zur Zusammenarbeit für spezielle Projektmanagementmethoden dienen, z. B. Agilo for Scrum von der Firma agile42, die sich auf die Zusammenarbeit in Scrum-Teams spezialisiert haben.

Uta Kapp arbeitet als freiberufliche IT-Beraterin und systemischer Coach. Mit einer Kombination aus Fachberatung und Prozessberatung für Softwareprojekte hilft sie Entwicklungsteams bei der Bewältigung der ständig steigenden Komplexität. Hier kommen die agile Softwareentwicklungsmethode Scrum und Organisationsaufstellungen zum Einsatz. Der Einsatz von Collaboration-Plattformen ist ein weiterer Schwerpunkt.

Jean Pierre Berchez ist Geschäftsführer der HLSC UG und beschäftigt sich seit mehr als 15 Jahren mit den Themen Projektmanagement, Software Engineering und objektorientierte Softwareentwicklung. In den letzten Jahren liegt sein Interesse insbesondere auf den Themengebieten agile Entwicklung mit Schwerpunkt Scrum sowie "Collaborative" Software Development. Neben seiner Tätigkeit als Geschäftsführer ist Jean Pierre Berchez auch als Lehrbeauftragter an den BAs Stuttgart und Heidenheim sowie an der Hochschule Liechtenstein für die Themen "Anforderungsmanagement", "Scrum" und "Collaborative Software Development" tätig. Er organisiert unter anderem den Community-Event "Scrum-Day" in Deutschland.

Original Artikel auf:

http://it-republik.de/dotnet/artikel/Kommunikation-zwischen-Entwicklern-Benutzern-und-Management-3051.html

---

andere Artikel dieser Serie

- Kollaborieren mit Plattform
- Integrationen und Schnittstellen – "Wie es Euch gefällt"
- Bughunting
- Was macht ein Open Source-Projekt erfolgreich?
- Metriken, Statistiken, Dashboards
- Wikinomics
- Kollabieren oder Kollaborieren
- Integration statt Brückenbau
- "Projekt Gold" oder: Der Weg führt zum Ziel