

## Bug hunting

(Link zum Artikel: <http://www.it-republik.de/dotnet/artikel/3121>)

### Fehlerdatenbanken fördern Benutzerfeedback

Text: von Uta Kapp und Jean Pierre Berchez

Nur wenn es für Benutzer einfach ist, Bugs zu melden, werden sie dem Entwicklungsteam das nötige Feedback liefern, um effizient voranzukommen. Jedes Softwarerelease hat seine Bugs. Jeder "bekannte" Bug ist schon fast behoben. Damit es für den Benutzer attraktiv ist, Fehler zu melden, wird idealerweise eine zentral zugängliche, einfach bedienbare Fehlerdatenbank für ein Projekt bereitgestellt.

Tracker sind das Herzstück einer jeden Collaboration-Plattform. Sie sind im Kern eine Liste mit zu erledigenden Aufgaben. Die ersten Tracker gab es zur Verwaltung und Verfolgung von Hinweisen auf Fehler und sie wurden deshalb "Bug Tracker" genannt. Bugzilla ist eine der ersten Tracker, die für viele Open-Source-Projekte im Einsatz ist. Mit der Zeit stellte sich heraus, dass Tracker nicht nur für die Verfolgung von Fehlern nützlich sind. So entwickelten sich Tracker zu einer eigenen Kategorie von Tools zur Verwaltung von Aufgaben. Diese Aufgaben werden in verschiedenen Plattformen unterschiedlich bezeichnet. Sie heißen Workitems, Issues, Services oder ähnlich. Jede dieser Aufgaben hat in seiner einfachsten Form ein Attribut, das die Aufgabe beschreibt und einen Status, z. B. offen oder erledigt.

Aufgaben können mithilfe von Verlinkungen zueinander in Beziehung gesetzt werden. Tracker sind adaptierbar. Das Grundprinzip bleibt gleich, aber die Attribute sind anpassbar. So kann für jede Art von Aufgabe ein eigenes Profil erzeugt werden. Das ist wie ein eigener Datentyp. Es gibt Collaboration-Plattformen mit sehr ausgefeilten Trackern, bei denen ganze Hierarchien von Aufgaben abgebildet werden können. Wenn ein Tracker eine Workflow Engine enthält, dann lassen sich vorgegebene Arbeitsabläufe realisieren. So kann die Veränderung des Status einer Aufgabe z. B. Folgeaufgaben und eine Benachrichtigung von Personen auslösen.

### Bug Tracking

Nach wie vor ist der Bug Tracker der wichtigste Tracker in jedem Softwareentwicklungsprojekt. Dieser Tracker verwaltet die gemeldeten Fehler. Hier ist ablesbar, welche Defekte in Bearbeitung sind, welche Bugs erledigt und getestet wurden und in welches Release die Korrektur veröffentlicht wird. Der Bug Tracker sollte für alle Tester und Benutzer eines Projekts zugänglich sein, sodass Fehler unbürokratisch gemeldet werden können. Es muss einfach sein, Fehler zu melden. Er enthält mindestens eine Beschreibung des Fehlers und den Status *offen*, *getestet*, *wird nicht repariert* oder *geschlossen*.

Weiterhin sind Informationen darüber enthalten, wer den Fehler wann gemeldet hat und wer die Aufgabe übernimmt, den Fehler zu beheben. Ein diszipliniert geführter Bug Tracker gibt Auskunft darüber, wie viele Fehler in einem Projekt aktuell offen sind und wie schnell im Schnitt ein Fehler korrigiert wird. Tracker ermöglichen es, zu einzelnen Aufgaben eine direkte Kommunikation zu führen und Kommentare abzugeben oder sich Benachrichtigungen senden zu lassen, wenn der Status des Fehlers sich ändert.

## **Anforderungs- und Änderungs-Tracker**

Anforderungs-Tracker sind ähnlich aufgebaut wie Bug Tracker. Sie enthalten Informationen zu Features, die neu implementiert werden sollen. Oft stellt sich heraus, dass ein Fehler eigentlich ein neues Feature ist. Dann wandert der Bug in den Anforderungs-Tracker. Auch Änderungen müssen sehr fein von Fehlern unterschieden werden. Um einen Bug neu als Änderungswunsch zu klassifizieren, sollte das Verschieben von Workitems zwischen Trackern möglich sein.

Anforderungs und Änderungs-Tracker können zusätzlich z. B. Felder für die Beschreibung des Anwendungsfalls und den geschätzten Realisierungsaufwand haben. Da Tracker idealerweise eine Funktion für das Importieren von Daten enthalten, kann hier eine Verbindung zu Spezialtools aus dem Requirements Engineering hergestellt werden, z. B. zu einem UML-Tool.

## **Meilenstein-Tracker**

Tracker können beliebig angepasst werden, um damit verschiedene Aufgaben abzuwickeln. Nützlich für die Buchhaltung von freigegebenen, lauffähigen Softwareständen ist ein Meilenstein-Tracker. Hier stellt jedes Issue einen Meilenstein dar, mit Releasedatum, Releaseversionsnummer, Link auf Releasenotes, Build-Resultate, Tag-Name im Sourcecode-Archiv und einer Beschreibung. Dieser Tracker, mit allen Links, dient dann als Einstieg für die Nachvollziehbarkeit in einem Release. Der Bug Tracker steht wiederum im Bezug zu dem Meilenstein-Tracker, damit festgestellt werden kann, in welchem Release oder Patch ein Fehler repariert ist.

## **Prozessabhängige Tracker**

Das Vorgehen bei der Entwicklung von Software wird nach vielen verschiedenen Vorgehens- und Prozessmodellen gestaltet. Jedes Team verwendet entweder selbst definierte Prozesse oder richtet sich nach fest vorgegebenen Modellen. Tracker für Bugs und andere Anforderungen müssen für diese Prozesse anpassbar sein. Bekannte solche Prozessmodelle sind die agilen Vorgehensmodelle SCRUM und XtremeProgramming.

Prozesse können von einer Collaboration-Plattform mithilfe von Trackern unterstützt werden. SCRUM ist eine iterative, agile Methode, bei der jede Iteration eines Produkts ein „30-Tage-Sprint“ genannt wird. Bei SCRUM wird mit so genannten "Backlogs" gearbeitet. Ein Sprint-Backlog ist z. B. eine Liste mit Features, die in einem Sprint implementiert werden soll. Solch ein Backlog wird auf einer Collaboration-Plattform als Backlog Tracker abgebildet. Die Fehler aus dem Bug-Tracker fließen hier auch mit ein. Er enthält Felder für Beschreibung, geschätzte Aufwandspunkte und erledigte Aufwandspunkte. Wenn die Software des Trackers sich auch Historiendaten merkt, dann lässt sich ein Burn Down Chart hieraus erzeugen. Das ist eine Metrik, die den Projektfortschritt abbildet, in dem sie die offenen Arbeitseinheiten täglich darstellt.

In dem Xtreme-Programming-Prozessmodell, werden z. B. Anforderungen des Kunden als Story Cards dargestellt. Das sind Beschreibungen der Kundenanforderungen in Form von Geschichten. Ein Tracker kann so angepasst werden, dass er als Story Card Tracker dient. Jedes Workitem stellt hier eine einzelne User-Story dar und kann etwa für Meetings auf

Papier ausgedruckt werden. Da Tracker an das Projekt anpassbar sind, lässt sich damit fast jedes Prozessmodell umsetzen.

*Uta Kapp arbeitet als freiberufliche IT-Beraterin und systemischer Coach. Mit einer Kombination aus Fach- 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/Bughunting-3121.html>

---

#### andere Artikel dieser Serie

- [Kollaborieren mit Plattform](#)
- [Integrationen und Schnittstellen – "Wie es Euch gefällt"](#)
- [Was macht ein Open Source-Projekt erfolgreich?](#)
- [Metriken, Statistiken, Dashboards](#)
- [Kommunikation zwischen Entwicklern, Benutzern und Management](#)
- [Wikinomics](#)
- [Kollabieren oder Kollaborieren](#)
- [Integration statt Brückenbau](#)
- ["Projekt Gold" oder: Der Weg führt zum Ziel](#)