Test Mobs –
Unsere Erfahrungen mit gemeinschaftlichem Testen von mobilen Apps

Simon André Scherr, Frank Elberzhager, Adeline Silva Schäfer
Fraunhofer IESE, Fraunhofer-Platz 1, 67663 Kaiserslautern
{simon.scherr, frank.elberzhager, adeline.schaefer}@iese.fraunhofer.de

Zusammenfassung:
Qualitätssicherung ist ein fester Bestandteil der Softwareentwicklung. Neben fest etablierten Methoden wie Dokumentenreviews oder Unit- und Systemtests werden immer wieder mal leichtgewichtigere Ansätze in Ergänzung genutzt. Dazu zählen auch sogenannte Test Mobs, in welchem eine Handvoll Teilnehmer zusammen nach Problemen am Softwareprodukt suchen. Wir haben mittlerweile 14 Test Mobs während der Entwicklung von Apps für den ländlichen Raum durchgeführt und berichten von Anpassungen und Erfahrungen beim Einsatz der Methode.

Schlüsselworte: Qualitätssicherung, Mobile Apps, Agile Methoden

1. Einleitung und Motivation

Software Qualitätssicherung ist so alt wie die Softwareentwicklung selbst: Seit Jahrzehnten kennen wir eine Vielzahl etablierter Review- und Testmethoden. Erste Veröffentlichungen zu dem Thema gab es beispielsweise von Myers 1979 [1] zu Testmethoden, oder zu Inspektionen von Fagan 1976 [2]. In der Folgezeit entwickelten sich Methoden weiter und wurden u.a. an verschiedene Kontexte und Entwicklungsmethoden angepasst.

Insbesondere in zunehmend agilen Umfeldern ab dem Jahr 2000 wurden etablierte Methoden oftmals durch leichtgewichtige Ansätze unterstützt, beispielsweise durch „smoke testing“ oder exploratives Testen. Eine weitere Methode ist der sogenannte Test Mob. Es handelt sich dabei um eine Art Gruppenreview des Testobjekts, welches unter anderem Aspekte des agilen Testens aufgreift (z.B. das explorative Vorgehen, ein kollaborativer Modus oder auch design thinking (Ideation, Kreativität)). Wesentliches Ziel ist es, gemeinsam in kurzer Zeit einen Teil des Softwareprodukts zu prüfen und Fehler bzw. Auffälligkeiten zu finden.

In diesem Beitrag möchten wir unsere Erfahrung mit Test Mobs schildern und damit auch andere Entwickler und Tester einladen, diese Methode für sich zu entdecken. Während der Entwicklung unterschiedlicher mobiler Apps haben wir den Test Mob 14 Mal seit Januar 2019 eingesetzt. Wir haben den Prozess stetig an unsere eigenen Bedürfnisse angepasst und eine Vielzahl von Problemen damit gefunden. Wir skizzieren zunächst unseren Hintergrund der Softwareentwicklung und der Qualitätssicherung, erklären die Hauptaspekte des Test Mobs und schildern unsere Erfahrungen.

2. Hintergrund und Herausforderung

Aktuell forschen wir an der Frage, wie die Attraktivität des ländlichen Raums in Deutschland durch digitale Dienste gesteigert werden kann. Dazu haben wir, u.a. im Projekt Digitale Dörfer[1], in den letzten Jahren eine Plattform sowie unterschiedliche Dienste entwickelt. Ein solcher Dienst ist der sogenannte DorfFunk, mit dem sich Bewohner von Kommunen miteinander austauschen, Dinge tauschen oder auch Vorschläge an ihre Gemeinde senden können. Das Team ist klein und arbeitet auch nicht in Vollzeit an der App.

Neben der generellen Fragestellung, wie Nutzer für die Apps gewonnen werden können, stellen sich typische Qualitätssicherungsfragen wie beispielsweise: Wie kommen Nutzer mit der Usability und User Experience zurecht bzw. was haben sie für Anmerkungen; wie ist die Performance; wo treten unerwartete funktionale Fehler auf und wie wird die Robustheit hochgehalten?

3. Test Mobs in der Qualitätssicherung

Test Mobs greifen eines der Trendthemen aus der agilen Softwareentwicklung, nämlich „Mob Programming“, auf und versuchen es auf die Welt des Testens zu übertragen. Mob Programming erweitert Pair Programming um mehr als zwei Personen: Statt zu zweit vor einen Rechner zu sitzen, nimmt beim Mob Programming ein ganzes Team Teil, um Entwicklungsprobleme zu lösen [3]. Das Team versammelt sich dabei um einen ausreichend großen Bildschirm und versucht gemeinschaftlich mit nur einem Keyboard zu entwickeln. Das Team, oder der Mob, ist oftmals tatsächlich bunt gemischt:

[1] https://www.digitale-doerfer.de/

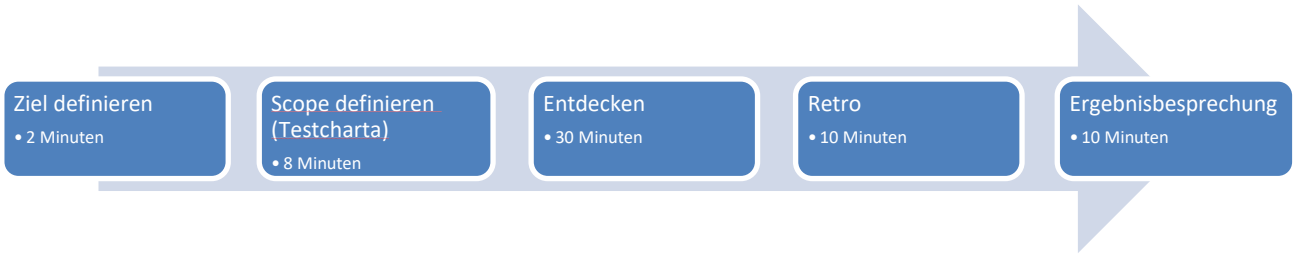

Abbildung 1. Phasen eines Test Mobs für DorfFunk sowie deren Dauer

Nicht nur Entwickler und Tester nehmen daran teil, sondern auch Architekten, Designer, Product Owner oder etwa der Scrum Master.

Mit Durchführung eines Test Mobs setzt man genau auf solche Prinzipien. Ein diverses Team versammelt sich, um gemeinsam in einer Testsitzung das Produkt zu testen. Zunächst setzt man das Gesamtziel für eine Session fest und erarbeitet danach eine Test-Charta oder Test-Mindmap. Dies dient zur Ermittlung eines möglichst trennscharfen Bereichs zwischen dem, was man in der Sitzung gemeinsam testen möchte und dem, was eben nicht betrachtet werden soll. Damit befindet sich ein Test Mob im Spektrum von vollständig detailliert ausgestalteten Testfällen und explorativem Testen in einem Mittelpunkt.

Daraufhin werden den einzelnen Teilnehmern Rollen (Driver, Navigator, Protokollant, Moderator) zugeteilt und die Entdeckerphase beginnt, für die ein striktes Zeitlimit festgesetzt wird. Aufgabe der Navigatoren ist die Definition der Schritte während der Testausführung, welche der Driver ausführt. Bei einer minimalistischen Verteilung von Rollen sind grundsätzlich alle Teilnehmer Navigatoren bis auf den Driver. Zusätzlich führt einer der Navigatoren das Protokoll über die Testsitzung. Während die Navigatoren jederzeit Vorschläge für nächste Schritte machen können, ist vom Driver Disziplin gefragt: Der Driver darf keine eigenmächtigen Interaktionen mit dem Produkt durchführen und muss sich stattdessen seine letzten Interaktionsschritte merken, um – falls einer der Navigatoren ein potentielles Problem feststellt – erklären zu können, was genau gemacht wurde und um die Schritte ggf. noch einmal zu wiederholen.

Jeder aus der Gruppe beobachtet genau die Software, die beispielsweise mittels eines Beamers projiziert wird. Sobald einer der Teilnehmer ein Problem feststellt, eine Umsetzung hinterfragt oder einen Change Request formuliert, schreibt der protokollführende Navigator diese auf. Wichtig ist, dass in der Gruppe zu diesem Zeitpunkt keine Diskussion über die Sinnhaftigkeit dieser Anmerkung geführt wird.

Nach Abschluss der Entdeckerphase werden die gefundenen Auffälligkeiten durchgegangen und Vorbereitungen getroffen, um entsprechende Tickets für das Backlog zu erstellen, welche dann vom Team bearbeitet werden.

Anschließend findet die Retrospektive statt. Hierbei kann jeder reihum sagen, was ihm am durchgeführten Test Mob gut gefallen hat oder welche Verbesserungsideen er für die folgenden Test Mobs hätte.

4. Unsere Erfahrungen mit Test Mobs

Im Folgenden beschreiben wir die Erfahrungen, die wir mit Test Mobs gemacht haben, aber auch, welche Justierungen vorgenommen wurden, um das Verfahren den Eigenheiten unseres Teams anzupassen.

4.1 Ein Test Mob für unsere App „DorfFunk“

Wir nutzen Test Mobs seit Januar 2019 alle zwei Wochen im Rahmen unserer regulären Qualitätssicherung für die App DorfFunk[2]. Der von uns angewendete Ablauf sowie die Längen der Phasen ist in Abbildung 1 dargestellt. Hierbei haben sich in unserem Fall eine Entdeckerphase von 30 Minuten sowie 6 Teilnehmer bewährt. Mit der anfänglichen Zieldefinition sowie dem Entwerfen der Testcharta, der Retro und der Ergebnisbesprechung kommt ein Mob bei uns auf eine Dauer von einer Stunde. In der Entdeckerphase achten wir auf alle auffallenden Details, weshalb sich schon nach wenigen Test Mobs herauskristallisierte, dass einer der Navigatoren mit expliziten Moderationsaufgaben betraut werden muss.

Dieser Moderator achtet darauf, dass keine Diskussionen in der Entdeckerphase geführt werden und dass der Umfang, der in der Test-Charta definiert wurde, abgearbeitet wird.

Neben der Notwendigkeit einer expliziten Moderatorenrolle ist es in unserem Fall nötig, den Protokollanten von seiner Rolle als Navigator zu befreien. Bei sehr ergiebigen Test Mobs kann sich der Protokollant so ganz

[2] https://www.digitale-doerfer.de/unsere-loesungen/dorffunk/

der Dokumentation der gefundenen Elemente widmen, was die Qualität der Notizen steigert und so hinterher die Erstellung besserer Tickets für den Entwicklungsprozess erlaubt.

Unsere Teilnehmer entstammen nicht immer nur dem Produktteam für die App: Regelmäßig an den Test Mobs nehmen auch Mitarbeiter von Nachbarteams, die das Produkt teilweise kennen, sowie produktentwicklungsfremde Personen teil. Beide Gruppen haben starkes Potential Dinge in der App zu hinterfragen, wobei die zweite Gruppe an Teilnehmern bei uns stärker in der Lage war, die Perspektive von Nutzern einzunehmen.

Daher hat es sich für unsere Mobs eingebürgert, dass wir die Retrospektive vor der Diskussion der gefundenen Elemente machen. Nach der Retrospektive verlassen alle, die nicht zum Produktkernteam gehören, den Mob und der verbleibende Rest erstellt Tickets, aktualisiert bereits bestehende Tickets und gibt eine erste Aufwandschätzung ab.

4.2 Was unsere Test Mobs aufzeigen

Schaut man sich die gefundenen Elemente nach einem Test Mob an, lautet die Einschätzung der Teilnehmer sehr häufig: „Das hätte ich nie alles alleine in so kurzer Zeit gefunden“. Dieses Feedback blieb über die letzten Monate bei den Test Mobs konstant. Durch die Teamdynamik werden Szenarien erzeugt, die sich ein einzelner Tester in diesen Kombinationen nicht ausdenken würde. Darüber hinaus achtet die Teamgemeinschaft immer wieder auf unterschiedliche Bereiche.

Durch den Mix an verschiedenen Perspektiven auf ein Produkt wird in der kurzen Zeit sowohl funktional als auch nicht-funktional getestet. Selbst sehr kleine Usability-Fehler oder Unstimmigkeiten fallen so dem Team schnell auf. Ein typischer Test Mob erlaubt es uns in einer halben Stunde 15-25 Auffälligkeiten für die Bewertungsrunde zu identifizieren. Hierbei muss man sich allerdings im Klaren sein, dass nicht alle diese Elemente Bugs sind, sondern auch Ideen für neue Funktionen sowie Änderungswünsche.

4.3 Was Test Mobs sonst noch bei uns bewirken

Ein interessantes Fazit, dass wir in jedem Test Mob feststellen konnten ist, dass den Teilnehmern der Mob Spaß macht. Damit ist das Testen in dieser Runde für keinen der Teilnehmer eine lästige Pflicht, sondern eine willkommene Abwechslung. Der Test Mob erlaubt es darüber hinaus dem gesamten Team das Produkt immer wieder kritisch zu hinterfragen und über einst bewusst getroffene Entscheidungen erneut zu reflektieren oder diese neueren Teammitgliedern zu erklären. Das trägt zu einem besseren Verständnis des Produktes über das ganze Team hinweg bei.

5. Fazit und Ausblick

Test Mobs haben in unserem Umfeld eindeutig für eine weitere Qualitätsverbesserung gesorgt, eine Vielzahl von vorher nicht bekannten Problemen wurde durch das gemeinsame Testen gefunden. Der Prozess ist leichtgewichtig und macht den Teilnehmern Spaß, zudem wird Wissen transportiert. Wichtig ist vor allem, dass der Prozess von Zeit zu Zeit reflektiert und an neue Bedürfnisse angepasst wird. In der Durchführung selbst gilt, ähnlich zu Gruppensitzungen bei Reviews, dass ein Moderator darauf achtet, dass der Fokus eingehalten wird und keine Zeit durch Nebendiskussionen verschwendet wird. Unter diesen Gegebenheiten können wir die Methodik als Ergänzung zu klassischen Qualitätssicherungsmethoden empfehlen. Wir gehen davon aus, dass wir noch weitere Anpassungen in unserem Umfeld nach Bedarf durchführen werden. Ein Beispiel hierfür könnte eine bessere Dokumentation und ggf. Klassifikation der gefundenen Probleme sein, so dass diese künftig noch besser ausgewertet werden können, was zu einer höheren Automatisierung von Tests führen könnte. Zudem wollen wir die Methodik auf weitere Apps ausweiten, die bei uns entwickelt werden.

Danksagung

Dieser Beitrag wurde im Kontext des Projekts EnStadt: Pfaff (Förderkennzeichen: 03SBE112D und 03SBE112G) erstellt, gefördert vom Bundesministerium für Wirtschaft und Energie (BMWi) sowie vom Bundesministerium für Bildung und Forschung (BMBF).

Referenzen

[1] G. Meyer, The Art of Software Testing, Wiley and Sons, 1979.

[2] M. E. Fagan, „Design and Code Inspections to Reduce Errors in Program Development,“ IBM Systems Journal, Bd. 15, Nr. 3, pp. 182-211, 1976.

[3] A.-C. Klose, „Vom Pair Programming zum Mob Programming – Programmiertechniken im Wandel,“ 13 06 20916. [Online]. Available: https://entwickler.de/online/agile/pair-programming-mob-programming-249211.html. [Zugriff am 18 07 2019].