Sonderdruck aus OBJEKTspektrum

Einfache UX-Methoden für agile Entwicklerteams

„Keep it simple and iterate!“

Die meisten Entwicklerteams widmen sich in ihren Unternehmen der Modernisierung oder kompletten Erneuerung von Softwareanwendungen. Die Digitalisierung bringt Veränderungen mit sich, die viele Bereiche ihrer täglichen Arbeit beeinflussen, wie Technologie, Architektur, Arbeitsabläufe und Formen der Zusammenarbeit. Auch Sichtweisen verändern sich: So ist eine wesentliche Veränderung der verstärkte Fokus auf den Endbenutzer und dessen „User Experience“ (UX).

Der Endbenutzer möchte sich in einer Anwendung nicht mehr einfach nur zurechtfinden, er möchten sich darin „wohlfühlen“. Frustrierende Erlebnisse, Grübeln über die Bedeutung von UI-Elementen, langes Suchen nach relevanten Inhalten, plötzliche Fehler, die mit seltsamen technischen Fehlernummern erklärt werden, sind da ein absolutes No-go. Softwareanwendungen, die bei den Anwendern im Ranking ganz oben stehen, haben eins gemeinsam: Sie stellen konsequent die Bedürfnisse des Endbenutzers in den Mittelpunkt.

Facebook, Spotify oder Netflix setzen heute die Maßstäbe für Benutzerqualität, die Benutzer nicht nur in Consumer-Produkten, sondern auch in B2B-Lösungen oder unternehmensinternen Softwareanwendungen erwarten. Die Superstars der digitalen Unternehmen verfügen über das entsprechende Mindset sowie über agile Arbeitsweisen und Methoden, um innovative Services und Produkte zu entwickeln. Sie haben diese Arbeitsweisen mitgestaltet, vielfach erprobt und als Blaupause und Best Practices für alle anderen Unternehmen in die Welt getragen.

In traditionellen IT-Abteilungen sieht es da natürlich etwas anders aus. Welche IT-Abteilung leistet sich schon spezialisierte UX-Designer oder -Researcher, um die Bedürfnisse der Endbenutzer zu identifizieren und darauf basierend Software zu erstellen? Stattdessen teilen sich meist Programmierer und Fachexperten die Verantwortlichkeiten: Programmierer schreiben Code und sollen gleichzeitig 1A-Benutzeroberflächen gestalten. Fachexperten vermitteln die Anforderungen und sollen gleichzeitig für eine 1A-Benutzererfahrung sorgen. Manchem Mitarbeiter wird schon bei dieser Vorstellung schwindelig.

Doch auch Teams mit kleineren Budgets und ohne teure UX-Designer können sich mehr auf ihre Benutzer ausrichten. Auch sie sind schon mit einfachen Mitteln in der Lage, die Bedürfnisse ihrer Anwender um einiges besser zu erfüllen als bisher.

Dazu stellen wir in diesem Artikel *fünf einfache und kostengünstige UX-Methoden* vor, die jedes agile Entwicklerteam sofort für sich nutzen kann. Denn eins ist klar: Jedes Unternehmen profitiert von der Zufriedenheit eines Endbenutzers, der bei seiner täglichen Arbeit gerne und dabei auch noch effizient mit seiner Software interagiert.

Den Methoden möchten wir noch eine etwas provokante Äußerung von UX-Guru Jeff Patton voranstellen, die bei einem Seminar in Berlin fiel:

“Without UX Designers you still have UX, but without people that code you have no software.”

UX-Methode 1: Proto-Persona

Häufig übergeben der Fachbereich und ein Anforderungsmanager Anforderungen an das Entwicklerteam. Meistens entstehen daraus User-Storys, die dem klassischen Muster „Als <Benutzer> …, weil” folgen. Aber einmal ehrlich: Wie viel und was wissen wir denn wirklich über den Benutzer, der so selbstverständlich in der User-Story referenziert wird?

Um herausragende Benutzererfahrungen zu entwickeln, braucht es im ersten Schritt ein tiefes Verständnis und Wissen über den oder die Endbenutzer. Dieses Verständnis und Wissen darf mit dem gesamten Team geteilt werden. So sprechen alle vom gleichen Endbenutzer und Miss-

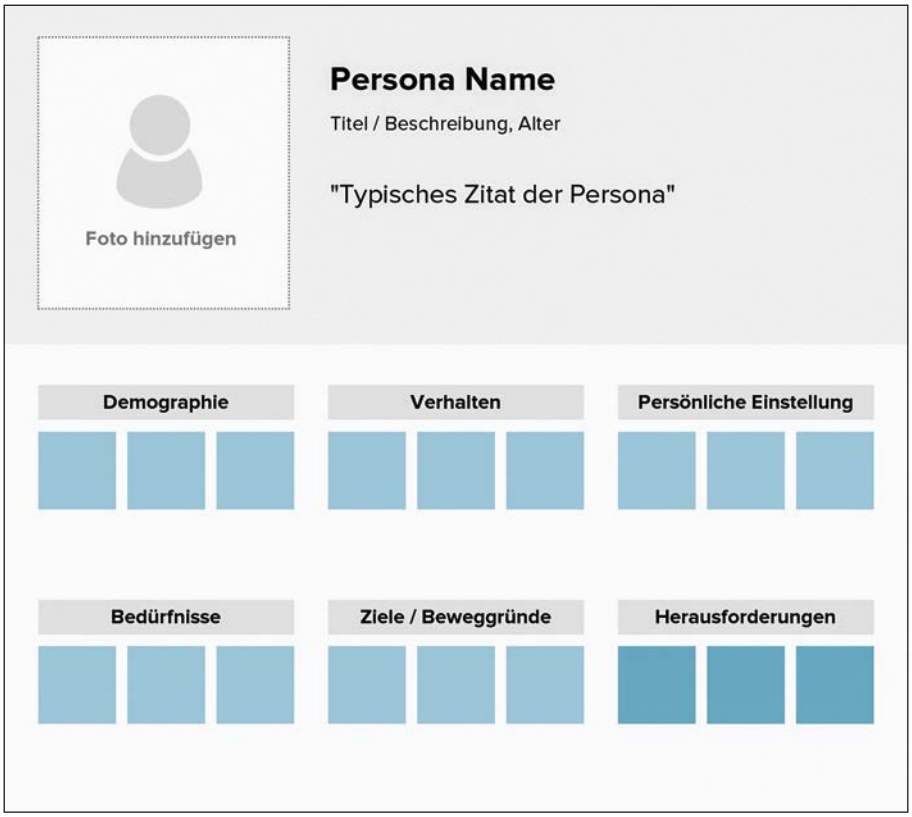


Abb. 1: Beispiel für ein Persona-Template

verständnissen wird vorgebeugt. Wenn jeder Einzelne in der Lage ist, die Welt mit den Augen der Endbenutzer zu sehen, fallen am Ende auch Design-Entscheidungen viel leichter.
Proto-Personas helfen, ein solches Verständnis zu entwickeln. Sie werden zu Beginn eines Projekts gemeinsam erstellt und im Verlauf der Produktentwicklung ständig mit neuen Erkenntnissen aus einfachen Endbenutzer-Tests wie Befragungen oder Beobachtungen angereichert.

Die einzelnen Schritte:

- Vorstellung und Erklärung des Persona-Templates (**vgl. Abbildung 1**).
- Eingrenzung der Zielgruppen und Auswahl der Benutzerprofile für die Personas, drei bis vier Personas reichen zu Anfang völlig aus. (Eine Persona ist immer besser als keine und trägt bedeutend zu besserer benutzerzentrierten Software bei.)
- Befüllen der Persona-Templates im Team.
- Diskussion der Personas und Verfeinerung.
- Aushang der Ergebnisse an zentraler Stelle im Projektraum.
- Kontinuierliche Verfeinerung der Personas im Entwicklungsprozess anhand von Testergebnissen usw.

Die (Proto-)Persona sollte für das Team immer präsent sein. Nur zu schnell wird im Laufe eines längeren Projekts vergessen, für wen die Software eigentlich entwickelt wird. Das Verständnis über die Endbenutzer, ihre Bedürfnisse und Eigenschaften sollte jedes Teammitglied verinnerlicht haben. Da sich die Persona durch Research und Benutzertests ständig verändert, sollte immer die aktuellste Version im Projektraum für alle sichtbar sein. **Abbildung 2** zeigt das Beispiel eines einfachen Persona-Templates, das in einem UX-Workshop vom Team befüllt wurde.

Abb. 2: Einfaches Persona-Template, das in einem UX-Workshop vom Team befüllt wurde

UX-Methode 2: User Story Map

Die User Story Map (**siehe Abbildung 3**) führt aus Sicht des Endbenutzers durch einen beispielhaften Anwendungsprozess (Story). Die Proto-Persona, die wir im ersten Schritt entwickelt haben, repräsentiert dabei den Endbenutzer. Wir empfehlen, die Story so einfach wie möglich zu erzählen. Währenddessen vermerkt der Moderator die wichtigsten Punkte auf „Sticky Notes" und klebt diese auf eine Tafel. So entsteht ein Gesamtbild der Story. Jeder im Team versteht, welche Story mit der Anwendung realisiert werden soll.

Die einzelnen Schritte:

- Referenzierung der Persona als Ausgangspunkt für die Geschichte.
- Verteilung von Sticky Notes und Stiften im Team.

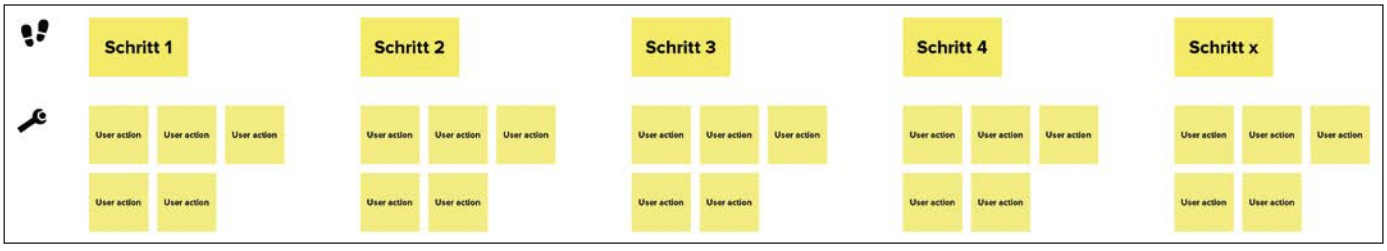


Abb. 3: Schematische Darstellung einer User Story Map

- Entwicklung der Geschichte aus Sicht der Persona.
- Festhalten auf Sticky Notes.
- Aufbringen der Sticky Notes an einer Wand oder Glasscheibe, sodass die Geschichte sichtbar wächst.
- Diskussion und Verfeinerung der Story Map.
- Definition jener Storys, die das Team weiterführen und als Prototyp konkretisieren möchte. (Bei der Auswahl dieser „wichtigsten" Storys sollte auf Metriken Bezug genommen werden, mit denen der Mehrwert und Erfolg gemessen werden kann.)

Abbildung 4 zeigt das Beispiel einer Story Map, die in einem UX-Workshop von Teilnehmern erstellt wurde, um Abläufe zu verstehen und Empathie für den Endbenutzer zu gewinnen.

UX-Methode 3: Papier-Prototypen

Papier-Prototypen lassen sich schnell erstellen und genauso schnell wieder verwerfen. Zwischen diesen beiden Polen entfalten sie ihre volle Kraft. So eignen sie sich, um einfache Benutzertests durchzuführen oder konkrete Design-Entwürfe und Abläufe im Team zu diskutieren. Die gewonnenen Erkenntnisse aus den Tests verfeinern den Prototyp. Dabei ist nicht viel Aufwand notwendig: ein neues Blatt Papier, ein Stift, eine einfache Skizze, fertig! Dabei geht es nicht um Perfektionismus – und genau das macht den Papier-Prototyp so effizient.

Papier-Prototypen (**siehe Abbildung 5**) eignen sich auch, um komplette Abläufe einer Benutzerführung zu testen. Der Testbenutzer navigiert dabei sozusagen im „Daumenkino" durch die Anwendung und das Team erfährt so aus erster Hand wichtige Informationen für die Verständlichkeit der entwickelten Benutzerführung.

UX-Methode 4: Heuristic Evaluation

Bei der Heuristic Evaluation handelt es sich um ein bewährtes Review-Verfahren aus den 1990er Jahren. Sie dient der Evaluierung und des Reviews von Benutzeroberflächen. Das ursprüngliche Konzept umfasst zehn Prinzipien. Beispiele für diese Prinzipien sind:

- *Visibility of system status:* Der Endbenutzer soll durch entsprechendes Feedback immer informiert sein, in welchem Zustand sich das System befindet. Fehlermeldungen, Hinweistexte oder visuelle Benachrichtigungen können helfen, den Systemstatus an den Endbenutzer zu kommunizieren.
- *Consistency and standards:* Besonders wenn Softwareanwendungen über die Zeit wachsen, leidet die Konsistenz der Benutzeroberfläche. Bezeichnungen etwa von Buttons werden nicht einheitlich eingesetzt, oder es gibt visuelle und sprachliche Brüche von Aktionen im gesamten Benutzerfluss. Das sorgt für Verwirrung sowie für eine hohe kognitive Belastung und Unsicherheit des Endbenutzers in der Interaktion mit der Anwendung.
- *Recognition rather than recall:* Eine Benutzeroberfläche sollte Dinge zeigen, die der Endbenutzer sofort erkennen kann (Recognition). Muss der Endbenutzer zuerst seine Erinnerung (Recall) aktivieren, um bestimmte Inhalte einer Anwendung zu verstehen und mit ihnen zu interagieren, bedeutet dies eine wesentlich höhere Belastung. Ein einfaches Beispiel für „Recognition" sind etwa „sprechende" grafische Icons, deren Bedeutung auf den ersten Blick erkannt wird. Ein Beispiel für einen

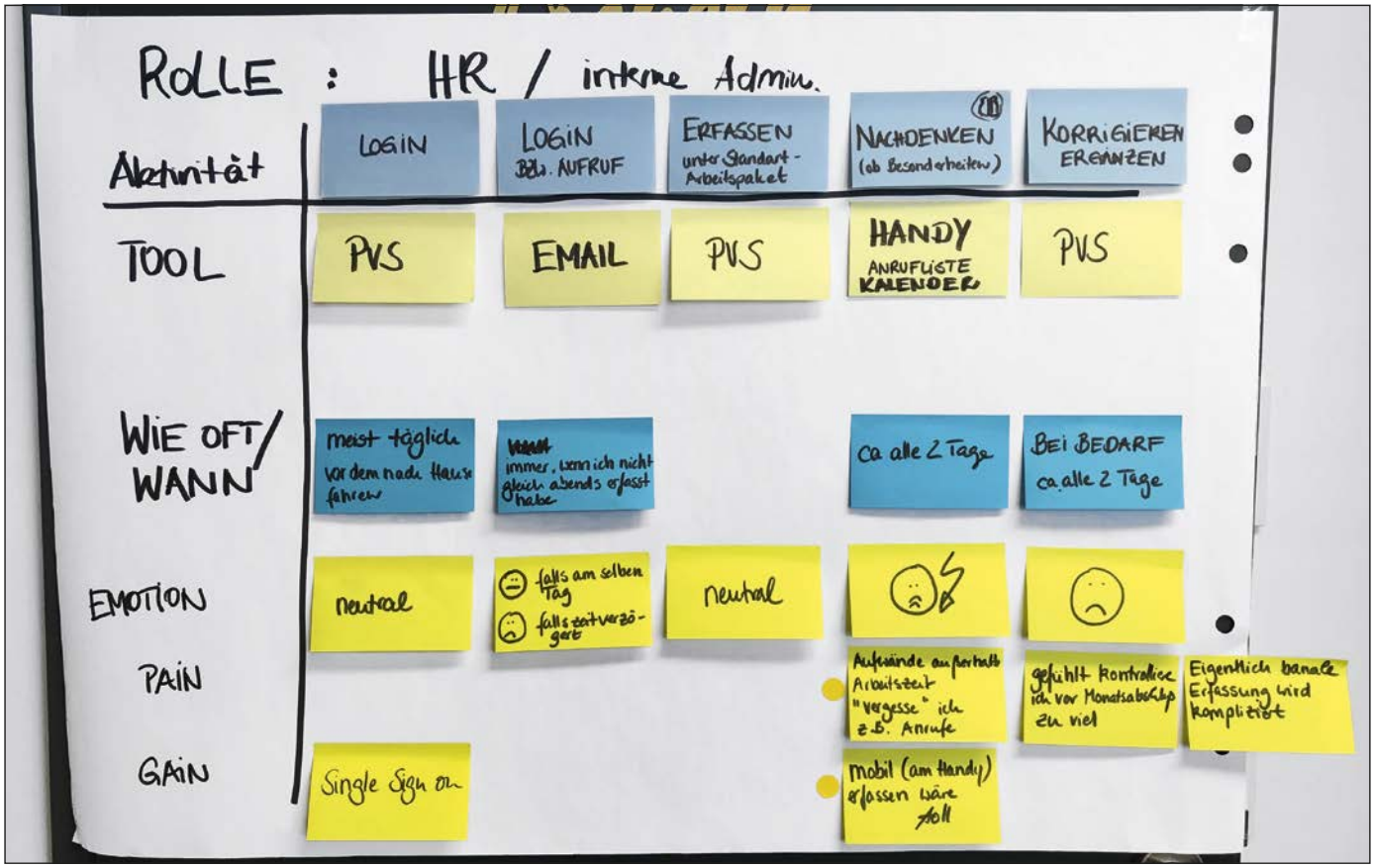


Abb. 4: Beispiel einer Story Map, die in einem UX-Workshop von Teilnehmern erstellt wurde, um Abläufe zu verstehen und Empathie für den Endbenutzer zu gewinnen

unnötigen Recall ist eine Login-Seite, auf der sich der Endbenutzer selbst an Benutzernamen und Passwort erinnern muss.

Die verbleibenden sieben Prinzipien lauten: “Match between system and the real world”, “User control and freedom”, “Error prevention, Flexibility and efficiency of use”, “Aesthetic and minimalist design”, “Help users recognize, diagnose, and recover from errors”, “Help and documentation”. (Quellenangabe: https://www.nngroup.com/articles/ten-usability-heuristics/)
Mithilfe dieses Prinzipienkatalogs kann das Entwicklerteam die Benutzeroberfläche strukturiert und vergleichbar bewerten. Wenn das Team es als sinnvoll erachtet, kann es den Katalog zusätzlich um eigene Prinzipien erweitern.

Die einzelnen Schritte einer Evaluierung:

- Vorbereitung eines einfachen Worksheets (**siehe Abbildung 6**).
- Definition der Bereiche des Anwendungssystems (System Locations), die getestet werden sollen. (Ein Bereich ist etwa eine Landingpage, der Einkaufskorb einer E-Commerce-Anwendung oder ein bestimmter Workflow, der in der Anwendung abgebildet ist.)
- Untersuchung des Systems auf Prinzipienverstöße.
- Dokumentation der Verstöße im Worksheet.
- Nach Abschluss der Dokumentation: Qualifizierung der Verstöße nach Schweregrad. (Die Skala reicht hierbei vom sogenannten „Cosmetic Problem“, „Minor Usability Problem“ über „Major Usability Problem“ bis zu „Usability Catastrophe“.)
- Kommunikation und Diskussion der Testergebnisse im Team.
- Gemeinsame Entscheidung über Korrekturen. (Die Entscheidung und Priorisierung sollte sich dabei am Schweregrad des Verstoßes orientieren.)

UX-Methode 5: „Think aloud“

„Think aloud“ ist eine klassische User-Research-Methode. Für jeden UX-Designer oder -Researcher ist diese Methode erste Wahl, wenn es darum geht, kostengünstig und effizient verwertbare Research-Ergebnisse zu erhalten, die das Produkt wirklich weiterbringen. Auch diese Methode ist einfach und schnell erlernbar. Wer ein paar Grundregeln beachtet, kommt schnell zu verblüffenden Erkenntnissen über die Erfahrungen, die der Endbenutzer des Produkts macht.

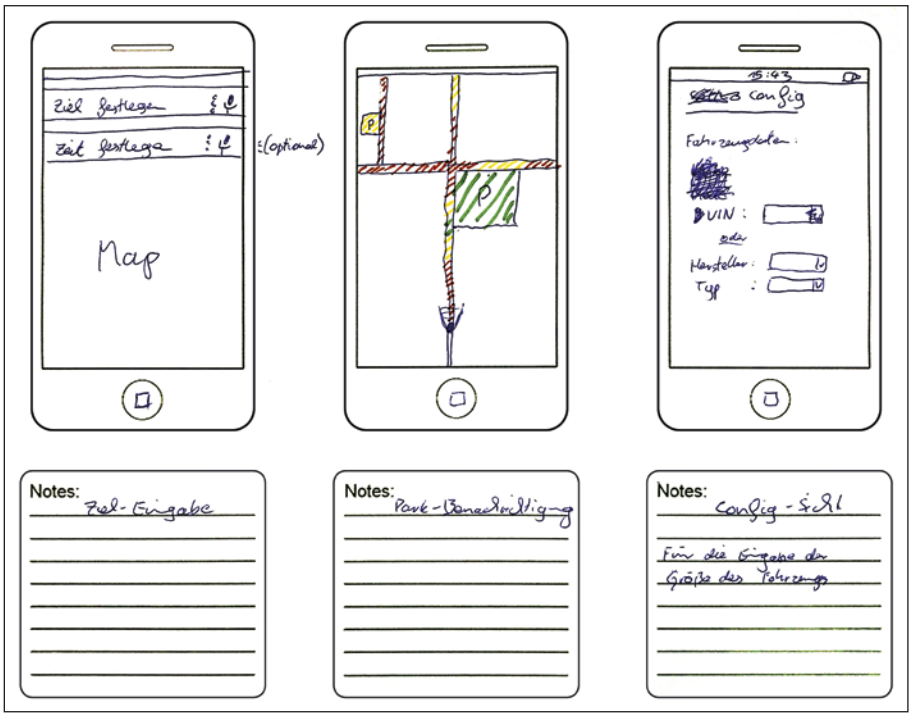
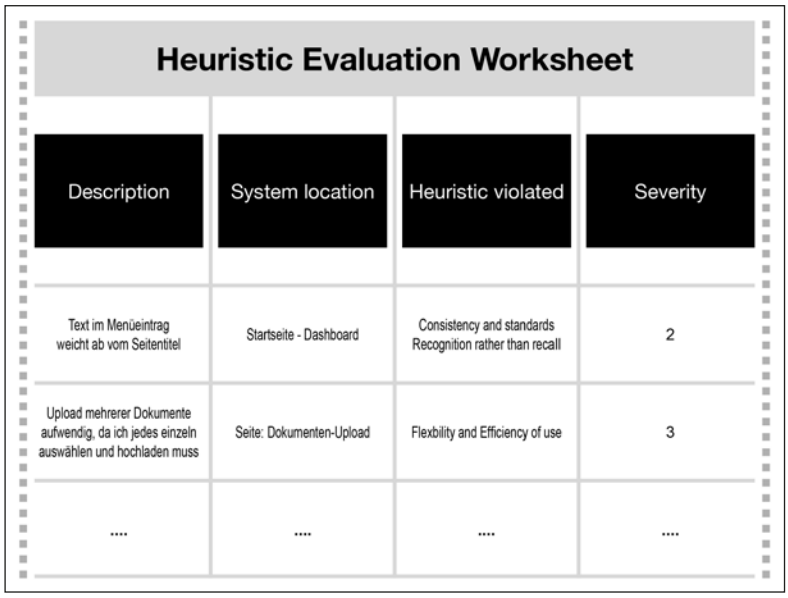


Abb. 5: Papier-Prototyp, der in einem UX-Workshop erstellt wurde, um UI-Design und -Abläufe im Team zu diskutieren

Das Grundprinzip von „Think aloud“ ist, dass die Teilnehmer laut aussprechen, was sie während der Durchführung von bestimmten Aufgaben mit der Software denken. Als Ergebnis gewinnt der Researcher oder Tester ein Verständnis dafür, warum der Benutzer bestimmte Aktionen ausführt, welche Reaktionen er zeigt oder welche Gedanken er sich über das Produkt macht.
Diese Ergebnisse werden im Anschluss ausgewertet. Sie sind positive Anmerkungen des Benutzers wie: „Diese Seite ist sehr übersichtlich und sieht aufgeräumt aus. Ich sehe auf einen Blick den Button, den ich benötige. Das ist sehr beruhigend.“ Am wertvollsten sind jedoch negative Anmerkungen, da sie Problemfelder und Verbesserungspotenzial aufdecken. Beispielsweise: „Das ist echt frustrierend. Der Button macht einfach nicht das, was ich erwarte. Am liebsten würde ich den Computer aus dem Fenster werfen.“ oder „Was ich hinter dem Button erwarten würde? Keine Ahnung, ich verstehe die Bezeichnung nicht. Das ist sehr verwirrend. Ich probiere es einfach aus. Aha. Ok. Naja. Das habe ich nun nicht erwartet.“
Gemeinsam leitet das Team aus den Erkenntnissen Aktionen und Konsequenzen ab, die festlegen, wie mit den identifizierten Problemen in den nächsten Sprints umgegangen wird und welche Verbesserungen in den Fokus der Implementierung rücken.

Die einzelnen Schritte zum Durchführen der „Think aloud“-Methode:

- Vorbereitung der Tasks, die mit dem Benutzer getestet werden sollen. Im Team sollten für den Test relevante Tasks bestimmt werden.
- Einladung einiger Test-Teilnehmer. Es reichen einige wenige Teilnehmer, die einzeln befragt werden. Bereits ab einer Anzahl von ca. sieben Teilnehmern ist der Erkenntnisgewinn so hoch, dass kaum eine Steigerung eintritt, wenn bedeutend mehr Teilnehmer befragt werden.
- Erstellen eines Interview-Leitfadens mit Einleitung, Hauptteil und Schluss.
- Vorbereitung der Testumgebung und Sicherstellung, dass die Testfälle auch hundertprozentig funktionieren.
- Durchführung der Tests mit ausführlichen Notizen. Daher lohnt es sich immer, den Test mit zwei Personen zu begleiten.
- Auswertung der Notizen. Am besten unmittelbar nach der „Think aloud“-Session. So sind die Eindrücke noch frisch und wenig verfälscht.
- Bewertung im Team und Entscheidung, welche Verbesserungen eingeplant werden sollen. Die Bewertung kann mithilfe einer Matrix zur Abwägung von Aufwand und Wichtigkeit erfolgen. Mithilfe der Matrix bewertet das Team die Erkenntnisse aus dem „Think aloud“-Test und hilft zu ent-



Heuristic Evaluation Worksheet

Description	System location	Heuristic violated	Severity
Text im Menüeintrag weicht ab vom Seitentitel	Startseite - Dashboard	Consistency and standards Recognition rather than recall	2
Upload mehrerer Dokumente aufwendig, da ich jedes einzeln auswählen und hochladen muss	Seite: Dokumenten-Upload	Flexbility and Efficiency of use	3
....	....	....	....

Abb. 6: Beispiel eines Worksheets für die Durchführung der Heuristic Evaluation

Abb. 7: Beispiel einer Matrix aus einem UX-Workshop

scheiden, welche Verbesserungen als Storys formuliert und umgesetzt werden.

- Im speziellen Fall der Matrix aus **Abbildung 7** wollte das Team die „Low Hanging Fruits" identifizieren. Die Elemente im obersten Quadranten bedeuten den geringsten Aufwand und die höchste Wichtigkeit. Das Team übernimmt diese Elemente, verfeinert sie und generiert daraus User-Storys für den Sprint.

Fazit

Die größte Stärke der vorgestellten UX-Methoden ist ihre einfache und unkomplizierte Anwendung. Sie lassen sich jederzeit spontan und ohne einen besonderen Aufwand für Vorbereitung oder Organisation einsetzen. Ihr größter Mehrwert für das Unternehmen liegt darin, die Ergebnisse gemeinsam im Team zu bearbeiten oder zumindest mit dem Team zu teilen und sichtbar zu halten. Die Artefakte, die bei ihrem Einsatz entstehen, dienen dabei als Anker und Referenz für Diskussionen und helfen im Arbeitsalltag, Probleme zu lösen oder neue Ideen in die Anwendung zu integrieren.

So werden die Artefakte zu Collaboration-Tools mit enormer Wirkung, die sich durch Einfachheit, Freiheit und Flexibilität auszeichnen. Immer getreu dem Motto „Keep it simple and iterate!" lässt sich auf diese Weise eine Win-Win-Win-Situation für Benutzer, Entwicklerteam und Unternehmen auf dem Weg in die Digitalisierung erreichen. ||

Der Autor

Andreas Lehner
(andreas.lehner@opitz-consulting.com) ist Trainer und Berater bei der OPITZ CONSULTING Deutschland GmbH. Er verfügt über mehr als 10 Jahre Erfahrung in der IT-Beratung von Großunternehmen. Seine Schwerpunkte sind die Schnittstellen von Mensch und Maschine sowie als wichtiges Fundament seiner Arbeit Design-Methoden und agile Ansätze.