Best Practices für die Entwicklung und den Test mobiler Anwendungen

BITKOM

Impressum

Herausgeber:	BITKOM Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. Albrechtstraße 10 A 10117 Berlin-Mitte Tel.: 030.27576-0 Fax: 030.27576-400 bitkom@bitkom.org www.bitkom.org
Ansprechpartner:	Manuel Fischer (BITKOM e.V.) Tel.: 030.27576-233 m.fischer@bitkom.org
Copyright:	BITKOM 2014
Redaktion:	Manuel Fischer (BITKOM)
Grafik/Layout:	Design Bureau kokliko / Matthias Winter (BITKOM)
Titelbild:	© Warakorn – Fotolia.de

Best Practices für die Entwicklung und den Test mobiler Anwendungen

Best Practices für die Entwicklung und den Test mobiler Anwendungen

BITKOM

Inhaltsverzeichnis

Mitwirkende:

- Manuel Fischer | BITKOM e.V.
- Dr. Marcus Iwanowski | Bluecarat AG
- Wolfgang Kunde | Bluecarat AG
- Dr. Frank Simon | Bluecarat AG
- Sascha Alpers | FZI Forschungszentrum Informatik
- Dr. Thomas Schuster | FZI Forschungs-zentrum Informatik
- Dr. Michael Gebhart | Gebhart Quality Analysis (QA) 82 GmbH
- Uwe Tewes | Gemalto GmbH
- Frank Maar | Microsoft Deutschland GmbH
- Friedrich Stahl | Testbirds GmbH
- Markus Steinhauser | Testbirds GmbH

1 Management Abstract

Die Verbreitung mobiler Geräte nimmt weiterhin kontinuierlich zu. Mittlerweile gibt es in Deutschland fast genauso viele Smartphones wie Menschen. Diese hohe Abdeckung bietet für viele Unternehmen ein großes Potential, mobile Technologien sowohl im B2B als auch im B2C-Bereich gewinnbringend einzusetzen.

Doch wie sehen eigentlich erfolgreiche mobile Anwendungen aus? Wie werden gute mobile Anwendungen erstellt? Sind mobile Geräte nur weitere Zielplattformen, die im Compiler einzustellen sind? Wie ändern sich organisatorische Parameter in der Erstellung?

Dieses Dokument möchte eine initiale Sammlung von Best Practices sein, die auf die Besonderheiten mobiler Anwendungen ganzheitlich eingehen und die im Kontext des BITKOM Kompetenzbereiches Software mit vielen beteiligten Organisationen entstanden sind. Hier ist insbesondere die inhaltliche Nähe zum Arbeitskreis Software-Engineering zu nennen. Die einzelnen Best Practices sind dabei nach einem einheitlichen Muster beschrieben und als Katalogform in die drei Themengebiete Produktspezifika, Prozessspezifika und Organisationsspezifika klassifiziert. Im vorliegenden Katalog werden diese Themengebiete als Katalogassets aufgeführt.

Aufgrund des relativ jungen Alters mobiler Technologien ist diese Best Practice Kollektion sicher noch nicht vollständig; allerdings hilft die Berücksichtigung der bisherigen 24 Best Practices schon sicher heute, einige große Risiken zu umschiffen.

Die hohe Volatilität mobiler Technologien wird auch an diesen Best Practices nicht spurlos vorbeiziehen: Von daher soll diese Sammlung gleichzeitig die Grundlage für eine lebendige Plattform sein, in der Experten und Interessierte sich austauschen, und weitere Best Practices formulieren, andere nachjustieren oder auch einige völlig entfernen. Diese Plattform ist aktuell unter der Internet-Seite http://software.bitkom.org/ zu hause. Dort ist auch jeweils die aktuellste Version dieses Best-Practice-Katalog zu finden.

Die Autoren würden sich freuen, wenn auf diese Art und Weise in Zukunft hochwertige mobile Anwendungen entstehen, die die Möglichkeiten mobiler Technologien wirklich gewinnbringend nutzen.

2 Einleitung

Das erste Automobil mit einem Benzinmotor hatte frappierende Ähnlichkeit mit einer Pferdekutsche. Aus heutiger Sicht wird dieses naive Einsetzen einer neuen Technik – in diesem Fall eines Benzinmotors – in eine existierende Infrastruktur – in diesem Fall z. B. das Chassis – häufig belächelt, da die wesentlichen Vorteile der neuen Technik gar nicht ausgenutzt werden konnten. Aber das ist mittlerweile über 100 Jahre her! Und so wundert es auch kaum, dass sich sowohl das Auto selbst, die Prozesse, die heute für die Erstellung von Autos verwendet werden, als auch die Organisation, die der Autoproduktion heute meist zugrunde liegt, massiv über die Jahrzehnte geändert haben: Ein Auto weist heute eben kaum noch Parallelen zu einer Kutsche auf. Die letzten Jahrzehnte haben eine Vielzahl von Best Practices für die moderne Automobilindustrie hervorgebracht, die die Besonderheiten der benzingetriebenen Fortbewegung bestmöglich berücksichtigen.

Das Problem dieser sehr bedächtigen, evolutionären Anpassung, die erst nach einigen Jahren oder Jahrzehnten die echten Mehrwerte einer Innovation nutzen lässt, gilt aktuell bei genauerer Betrachtung auch für die Entwicklung mobiler Anwendungen.

Auch im mobilen Bereich ist das Erkennen von Besonderheiten noch im »Kutschenstatus«, d. H. die gesamte Organisation, die verwendeten Prozesse und die erzeugten Ergebnisse orientieren sich jeweils noch stark an der Entwicklung klassischer Desktop-Applikationen. Dies beginnt bei vielen Reise-Applikationen mit der häufig immer noch notwendigen Eingabe des »VON-Ortes«, obwohl ein mobiles Endgerät weiß, wo es sich befindet, und endet beim klassischen »Schließen« einer Applikation, obwohl dies Freigeben von Speicher aufgrund der fehlenden Festplatte nicht nötig ist bzw. automatisiert vom Betriebssystem selbst durchgeführt wird.

Das Ziel dieses Dokumentes ist das strukturierte Sammeln von Best Practices, die bereits heute existieren und die mobilen Spezifika versuchen, möglichst ganzheitlich abzudecken:

- Das Produkt selbst: Wie sollten die Produkte gestaltet und die Software designed sein, um die durch mobile Technologien erreichbaren Mehrwerte effektiv nutzen zu können?
- Die Prozesse: Wie sollten die Prozesse, an deren Ende jeweils Produkte entstehen, aufgebaut sein, um die qualitativ hochwertigen Ergebnisse effizient entwickeln zu können?
- Die Organisation: Wie sollten Organisationen die Ressourcen, die in den Prozessen zum Einsatz kommen, entsprechend strukturieren, planen und auslasten, um die Prozesse möglichst leichtgewichtig, d. H. mit sehr kurzen Time-To-Market-Zeiten leben zu können?

Diese Sammlung an Best Practices ist als Netz von Warnungen und Empfehlungen zu verstehen. Werden diese berücksichtigt so steigt die Wahrscheinlichkeit, dass die entsprechende mobile Lösung effektive Mehrwerte beim Kunden erbringt; ein Risiko einer Fehlentwicklung kann damit aber in keinster Weise ausgeschlossen werden. Vielmehr stellt diese Sammlung eine Risikoreduktionstechnik dar, an dessen Ende allerdings immer noch ein Restrisiko existiert. Der Erfolg einer mobilen Lösung hängt insbesondere nicht nur von naturwissenschaftlich fassbaren Fakten wie einer hohen Usability und einer hohen Fehlerfreiheit ab, sondern auch von weichen Faktoren wie Markenwahrnehmung und Community-Abdeckung.

Werden die letzten 100 Jahre der kontinuierlichen Automobilweiterentwicklung betrachtet, so scheint es mutig, bereits 10 Jahre nach Aufkommen der ersten Smartphones einen solchen Katalog vorzustellen. Der Katalog hat daher heute auch noch keinerlei Anspruch auf Vollständigkeit, sondern soll nur den aktuellen Stand der Praxis wiederspiegeln. Die Erfahrung zeigt allerdings, dass bereits dieser initiale Zustand viele Fehlentwicklungen der jüngsten Zeit hätte verhindern können.

Darüber hinaus ist dieser Katalog explizit auf eine kontinuierliche Fortschreibung ausgelegt. Die Zukunft lässt sich auch heute noch nicht vorhersagen! Man kann nur versuchen, Änderungen möglichst früh zu erkennen und ggf. gegenzusteuern. Jeder ist daher herzlich eingeladen, sich selbst mit seinen Erfahrungen in diesem Bereich einzubringen.

Diese Best Practices wurden aus der Taskforce »Mobile App Quality« – einer Initiative des Arbeitskreises Software Engineering – heraus erarbeitet. Sowohl der Arbeitskreis als auch die TaskForce freuen sich jederzeit über engagierte Mitstreiter. Eine kurze E-Mail an Manuel Fischer vom BITKOM (m.fischer@bitkom.org) genügt.

3 Einführung in die Struktur des Kataloges

Um den Einsatz existierender Best Practices möglichst effizient durchführen zu können und gleichzeitig spätere, neue Best Practices strukturiert in den Katalog aufnehmen zu können, ist jede Best Practice des Katalogs nach ein und demselben Schema beschrieben. Im Folgenden werden Best Practices aufgeführt, die seitens der Mitwirkenden als relevant identifiziert wurden. Dabei sind die Best Practices gemäß der nachfolgenden Struktur beschrieben.

3.1 Asset

Auf der obersten Ebene wird das Artefakt, auf das sich die Empfehlung bezieht, als sogenanntes Asset klassifiziert. Entlang des EFQM-Frameworks werden die folgenden drei Assets unterschieden:

- Produkte stellen das End- oder Zwischenergebnissen von Arbeitsprozessen dar. Im folgenden werden hier vor allen Dingen Entwicklungsprodukte betrachtet, also die fertige Applikation, der erstellte Quelltext oder die entworfene Architektur.
- Prozesse beschreiben das Netzwerk von Aktivitäten mit verantwortlichen Rollen, die nötig sind, um die späteren Produkte zu erzeugen. Im Folgenden werden hier vor allen Dingen Entwicklungs-, Test- und Updateprozesse angesprochen.
- Organisation beschreibt das Netzwerk involvierter Personen, Rollen und Parteien, über die sich der Prozess zur Erstellung von Produkten definiert.

Da aus Endanwendersicht das Asset Produkt besonders wichtig ist, wurden hier bisher auch die meisten Best Practices identifiziert. Als weiteres Klassifikationsschema dient in diesem Bereich die ISO 25010, die einige Produkt-Qualitätsmerkmale aufführt. In diesem Katalog finden sich für das Produkt-Asset Empfehlungen für die folgenden Qualitätsmerkmale, deren englische Bezeichnungen aufgrund der Ausrichtung an ISO 25010 beibehalten wurden:

- Usability, also die Eignung des Gerätes, eine Aufgabe effizient und effektiv durchführen zu können. Gerade im mobilen Bereich wird hierunter insbesondere auch der Spaßfaktor (Joy of Use) subsumiert.
- Efficiency, also der bewusste Umgang mit zur Verfügung stehenden Ressourcen. Dies sind im mobilen Kontext vor allen Dingen Bildschirmplatz, Speicherplatz, CPU-Aktivität und Netzbandbreite.
- Security, also die Sicherstellung, dass Daten nur für solche Personen zugreifbar und ggf. änderbar sind, für die sie auch gedacht sind.
- Interoperability, also die Fähigkeit, für den Zweck der Anwendung sinnvolle und notwendige andere Systeme und Komponenten effektiv anzuschließen.
- Portability, also die Fähigkeit, ein Produkt für eine Zielplattform X mit möglichst wenig Aufwand auch für eine andere Zielplattform Y verfügbar zu machen.

3.2 Name & Beschreibung

Jede Best Practice hat einen eindeutigen Namen. Dieser ist als kurzer, einprägsamer Merksatz formuliert, so dass jeweils ungefähr offenkundig wird, worum es geht. Der Name ersetzt allerdings in keinem Fall die folgende detaillierter Beschreibung.

Mehrwert, Gültigkeit und Priorisierung

Das Ziel jeder Best Practice ist sicherzustellen, dass die mobile Technik möglichst gewinnbringend eingesetzt wird. Der damit angestrebte Mehrwert wird unter dem Punkt Mehrwert aufgeführt. Da der Mehrwert allerdings häufig nicht generell erreicht werden kann, sondern von spezifischen Rahmenparametern wie z. B. dem konkreten mobilen Betriebssystem oder spezifischen Handy-Layout abhängt, werden diese einschränkenden Faktoren unter Gültigkeit aufgeführt. Die folgende Priorisierung gibt dann noch an, wie relevant der Mehrwert für den Erfolg einer mobilen Lösung ist und wie direkt die Kausalkette zwischen Mehrwert und der Best Practice ist.

Referenzen und Beispiel

Die in diesem Leitfaden hinterlegten Best Practices basieren auf einer Vielzahl unterschiedlicher Quellen. Dies können Erfahrungen der am Leitfaden mit gearbeiteten Unternehmen, Fragmente anderer Veröffentlichungen oder gar Standards und Normen sein. Unter Referenzen werden die bekannten Verweise zu ähnlichen Hinweisen gegeben. Zum Abschluss eines jeden Katalogeintrags folgt ein kurzes, typisches Beispiel, anhand dessen die Best Practice und der angestrebte Mehrwert konkret erläutert wird.

4 Katalogasset: Produkt

4.1 Usability

#101 Ermögliche Steuerbarkeit mit »Zurück«-Button.

Bei mobilen Applikationen hat sich insbesondere aufgrund der eingeschränkten Display-Größe ein dialog-basierter Ansatz etabliert, bei dem nur die aktuelle bearbeitete Aufgabe als Dialog dargestellt wird. Anwender sind daher gefordert, häufig zwischen verschiedenen Dialogen zu wechseln, weshalb dieser Wechsel mit den üblichen Aktionen möglich sein muss. Diese grundsätzliche Forderung nach »Steuerbarkeit« inkludiert insbesondere auch die Möglichkeit, von einem geöffneten Dialog wieder zum vorherigen Dialog zurückzukehren. Hierfür haben verschiedene Hersteller jeweils spezifische Lösungen entwickelt. So wird die Rückkehr zum vorherigen Dialog entweder mit einem Software-Button oder mit einem Hardware-Button ermöglicht. Die Anwendung muss dabei die innerhalb der genutzten Plattform übliche Methode des »Zurück-gehens« unterstützen.

Mehrwert: Die Usability und dort als Unter-Anforderung die Steuerbarkeit spielt heutzutage bei Software eine zunehmende Rolle. Es ist für den Erfolg einer mobilen Anwendung daher essenziell, dass sie sich an die geltenden Vorgaben hält und somit nahtlos in die bestehende Plattform einfügt. Durch die einheitliche Nutzung einer entsprechenden Schaltfläche, um zu vorherigen Dialogen zurückzukehren, ist der Anwender mit der Software schneller vertraut.

Asset: Produkt

Gültigkeit und Priorisierung: Ein »Muss» für jede Anwendung in jeder Domäne und auf jeder Plattform. Bei nativen Anwendungen erfolgen hier bereits entsprechende Vorgaben durch die Plattform, so dass in diesen Fällen eine Einhaltung dieser Best Practice implizit erfolgt. Es gilt daher insbesondere bei mobilen Web-anwendungen, bei denen ein höheres Maß an Freiheit existiert, auf die Einhaltung dieser Best Practice zu achten.

Referenzen: Durch die Beeinflussung der Usability als essenzielle Qualitätseigenschaft für Software, kann ein Bezug zu ISO 25000 hergestellt werden. Das vorgestellte Kriterium Steuerbarkeit bildet somit eine Untereigen-schaft für die Usability.

Die Best Practice ist außerdem von der App Quality Alliance beschrieben.

Beispiele: –

#102 Nutze alle Möglichkeiten des Gerätes angemessen.

Smartphones bieten eine Vielzahl an technischen Möglich-keiten. Neben der Funktionalität des Telefonierens und der Verbindung zum Internet bieten weitere Schnitt-stellen und Sensoren viel Raum für neue Funktionen oder zur Verbesserung bereits vorhandener Implementierun-gen, indem man die gegebenen Möglichkeiten sinnvoll miteinander kombiniert.

Mehrwert: Die optimale Nutzung der zur Verfügung stehenden technischen Möglichkeiten stellt einen erheblichen Mehrwert für den Endanwender bereit und kann oft auch als Alleinstellungsmerkmal dienen.

Asset: Produkt

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

- Kombiniert man beispielsweise den Bewegungsensor mit dem GPS Modul, so kann man die GPS Koordinaten bspw. erst abfragen, wenn seit der letzten Abfrage das Gerät bewegt wurde. Hierdurch kann Energie gespart werden.

- Beobachtet man den Anwender mit der Kamera, kann man verhindern, dass das Smartphone in den Standby-Modus wechselt, während der Anwender liest. Samsung hat diese Funktionalität bereits in neuen Android-Smartphones umgesetzt, bezeichnet als Smart Stay.

- Beobachtet man die GPS Koordinaten, so kann man die WLAN-Verbindung automatisch aktivieren, wenn sich der Anwender in der Nähe eines für ihn freigeschalteten WLAN Netzwerk befindet (das benötigt weniger Energie, als laufend nach verfügbaren Hotspots zu suchen)

#103 Ermögliche die Verwendung der Anwendung in verschiedenen Geräteausrichtungen.

Anwender sind es heutzutage gewohnt, ein mobiles Gerät sowohl im Hoch- als auch im Querformat zu nutzen. Die Anwendung sollte daher in der Lage sein, beide Varianten zu unterstützen. Hierdurch kann der Nutzer abhängig von den jeweils angezeigten Inhalten die geeignetere Form wählen und somit die Anzeige der Inhalte je nach Bedarf optimieren.

Mehrwert: Der Anwender kann abhängig von den Inhalten zwischen einer Darstellung im Hoch- oder im Querformat unterscheiden und somit die Darstellung je nach Bedarf optimieren. Des Weiteren verhält sich die Anwendung dem derzeit gewohnten Muster, was letztlich die Akzeptanz der Anwendung erhöht.

Asset: Produkt

Gültigkeit und Priorisierung: Ein »Muss« für jede Anwendung in jeder Domäne und auf jeder Plattform.

Referenzen: Durch die Beeinflussung der Usability als essenzielle Qualitätseigenschaft für Software, kann ein Bezug zu ISO 25000 hergestellt werden. Das vorgestellte Kriterium bildet somit eine Untereigenschaft für die Usability.

Die Best Practice ist außerdem von der App Quality Alliance beschrieben.

Beispiele: –

#104 Stelle die Fingertauglichkeit der Anwendung sicher.

Ist nicht explizit vorgesehen, dass die Anwendung nur durch einen Stylus bedient werden soll, müssen Elemente groß genug sein, so dass sie mit dem Finger bedient werden können. Dies bedeutet u.a., dass Schaltflächen groß genug dargestellt werden und sich zwischen einzelnen Elementen genug Platz befindet, um versehentlich falsche Eingaben zu reduzieren. Selbst wenn heutzutage ein Tablet mit einem Stylus ausgeliefert wird, erfolgt die Bedienung in den meisten Fällen nicht ausschließlich über den Stylus. Es gilt also immer zu unterscheiden, ob eine Funktionalität innerhalb einer Anwendung explizit für den Stylus ausgelegt ist (bspw. die Malfunktion einer Anwendung) oder ob diese Funktionalität auch mit den Fingern bedient werden soll. Ggf. muss innerhalb einer Anwendung zwischen diesen beiden verschiedenen Methoden unterschieden werden.

Mehrwert: Die Anwendung folgt den Gewohnheiten des Anwenders. Wenn er es gewohnt ist, eine bestimmte Funktionalität mit den Fingern zu bedienen, so stellt ihm die Anwendung diese Funktionalität auch in geeigneter Form dar. Der Anwender kann somit in gewohnter Weise mit der Anwendung und dem Gerät interagieren und letztlich die Funktionalität der Anwendung nutzen. Hierdurch erhöht sich die Akzeptanz der Anwendung, da

sie in komfortabler Form bedient und somit überhaupt genutzt werden kann.

Asset: Produkt

Gültigkeit und Priorisierung: Ein »Muss« für jede Anwendung in jeder Domäne und auf jeder Plattform. Bei nativen Anwendungen wird dies meist bereits durch die genutzte Plattform vorgegeben. Es gilt daher, diese Best Practice insbesondere bei Anwendungen wie mobilen Webanwendungen zu berücksichtigen, in denen durch die Plattform nicht bereits entsprechende Vorgaben existieren und somit ein deutlich größerer Freiraum besteht.

Referenzen: Durch die Beeinflussung der Usability als essenzielle Qualitätseigenschaft für Software, kann ein Bezug zu ISO 25000 hergestellt werden. Das vorgestellte Kriterium bildet somit eine Untereigenschaft für die Usability.

Diese Best Practice ist außerdem von der App Quality Alliance beschrieben.

Beispiele: –

#105 Informiere den Anwender durchgehend über den Status der Applikation.

Mobile Anwendungen werden in den meisten Fällen nebenher genutzt, haben also nicht die volle Aufmerksamkeit des Anwenders. Zudem bieten Smartphones viele Funktionen/Schittstellen und unterliegen somit einer größeren Komplexität als normale Internet- oder Desktop-Anwendungen. Diese größere Komplexität drückt sich auch in einer höheren Fehleranfälligkeit aus. Die Fehler müssen dabei nicht zwingend in der Anwendung selbst liegen, sondern können auch durch die herrschenden Rahmenbedingungen hervorgerufen werden.

Mehrwert: Oft hat die laufende Applikation keinen Einfluss auf die gegebenen Umstände. Beispielsweise, wenn an einem bestimmten Ort keine gute Internetanbindung zur Verfügung steht, könnte der Anwender denken, das die Ursache des Problem in der Anwendung selbst liegt. Weist die Applikation allerdings den Anwender darauf hin, das die Applikation nicht wie gewohnt funktioniert, da die Verbindungsqualität zu schlecht ist, ist der Anwender gut informiert und wird die Anwendung nicht einfach deinstallieren.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

Beispiele: Der Anwender sollte z. B. über den Status der Internetverbindung informiert sein, damit er sich nicht wundert, wenn sich die Anwendung nicht wie gewohnt verhält.

Ebenso kann positives Feedback helfen, den Endanwender bei wichtigen Interaktionen zu unterstützen. Beispielswiese: »Datei wurde erfolgreich in der Cloud gespeichert.«

4.2 Efficiency

#106 Nutze externen Speicher.

Eine Vielzahl mobiler Geräte erlaubt es, den internen Speicher durch eine externe Speicherkarte zu ergänzen. Erzeugt die Anwendung größere Mengen an Daten (Musikstreaming etc.) und sieht das Gerät eine externe Speicherkarte (bspw. SD-Karte) vor, so muss die Anwendung die Möglichkeit bieten, die Daten auf der externen Speicherkarte abzulegen. Je nach Betriebssystem muss der Anwender hierbei frei darüber entscheiden können, ob er dies wünscht oder nicht. Des Weiteren ist darauf zu achten, dass die Anforderungen der Anwendung an den externen Speicher mit den technischen Möglichkeiten wie der Lese- und Schreibgeschwindigkeit übereinstimmen. Es sollten daher nur Daten auf den externen Speicher ausgelagert werden, die hierfür geeignet sind. Ggf. sollte der Anwender auf die Anforderungen an den externen Speicher hingewiesen werden.

Mehrwert: Der Anwender erhält hierdurch die Möglichkeit, den ergänzten Speicher des Gerätes zu nutzen. Hierdurch wird ermöglicht, dass die Anwendung auch auf Geräten nutzbar wird, die nicht ausreichend internen Speicher bieten. Letztlich wird hierdurch entschieden, ob die Anwendung überhaupt genutzt werden kann.

Asset: Produkt

Gültigkeit und Priorisierung: Ein »Soll« für jede Anwendung, die größere Mengen an Daten erzeugt, in jeder Domäne und auf jeder Plattform, ausgenommen Apple iOS, da diese Plattform keinen externen Speicher vorsieht.

Referenzen: Diese Best Practice ist auch von der App Quality Alliance beschrieben.

Beispiele: Android: Bei Android findet sich bspw. in den zentralen Einstellungen für viele Anwendungen die Möglichkeit, die Anwendung selbst auf einen externen Speicher zu verschieben. Diese Möglichkeit sollte unterstützt werden.

Des Weiteren bieten viele Anwendungen für Musikstreaming die Möglichkeit, die externe SD-Karte für die Daten zu nutzen.

Ebenso bieten viele Kamera-Anwendungen die Möglichkeit, die Fotos direkt auf der externen SD-Karte zu speichern.

#107 Minimiere den Datenverkehr.

Die Qualität der Internetverbindung eines mobilen Endgerätes unterliegt massiven Schwankungen. Damit die Qualität der Applikation dennoch durchgehend als hochwertig empfunden wird, muss man diesen Umstand berücksichtigen und die Abhängigkeit von der Qualität der Datenverbindung minimieren, indem man die benötigte Datenmenge so gering wie möglich hält. Zudem beeinflusst der Datenverkehr direkt den Energieverbrauch, die Performance, die Effizienz und somit auch die Kosten die durch die Anwendung einer Applikation entstehen können.

Mehrwert: Gute Anwendbarkeit aufgrund kurzer Antwortzeiten. Lange Lebensdauer des Akkus und somit eine hohe Verfügbarkeit. Die Applikation ist kein Kostentreiber aufgrund eines hohen eventuell unnötigen Datenvolumens.

Asset: Produkt

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: App Quality Alliance.

Beispiele:

- Beim Drehen des Gerätes (Wechsel von horizontaler auf vertikale Ansicht) nicht alle Daten neu laden
- Einsetzen eines lokalen Caches um das mehrfache Laden redundanter Daten zu vermeiden

- Pull-Lösungen für das Netzwerk optimieren. Es macht z. B. keinen Sinn, öfter per Pull nach einem neuen Status zu fragen, als es das gegebene Netzwerk erlaubt.

#108 Minimiere die Antwortzeiten.

Mobile Anwender sind extrem ungeduldig. Das liegt insbesondere an dem Kontext, in dem mobile Anwendungen überwiegend genutzt werden, denn sie werden meistens nebenbei genutzt und genießen somit nicht die volle Aufmerksamkeit des Anwenders. Wenn man aber eine Pause, die sich zufällig ergibt (Beispielsweise in einer Warteschlange), nutzt, um schnell einige Aufgaben mit dem Smartphone zu erledigen und die gewählte Anwendung nicht schnell / performant genug ist, um genau das in dem gegebenen Zeitrahmen zu tun, wird diese durch eine schnellere Anwendung ersetzt.

Mehrwert: In dem gegebenen Kontext gewinnt oft nicht die Anwendung mit den meisten Features, sondern die einfache intuitive Anwendung, die den Anwender in die Lage versetzt, unter den gegebenen Umständen (wenig Zeit und keine voll Aufmerksamkeit) eine Aufgabe zu erledigen.

Asset: Produkt

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: App Quality Alliance.

Beispiele: Beim Start einer Applikation werden nicht mehr als drei Sekunden toleriert, bis man ein Feedback erhält, ansonsten wird die App direkt wieder deinstalliert.

#109 Verwalte die Netzwerkverbindung aktiv.

Die Verbindung eines Smartphones mit dem Netzwerk kostet einerseits Energie (i.d.R. verbrauchen breitbandigere Verbindungstypen mehr Energie) und andererseits kostet der Transfer von Daten (engl. Traffic) in Abhängigkeit des Verbindungstyps Geld. Meist ist zusätzlicher Traffic über WLAN (ist man erst einmal, ggf. für eine bestimmte Nutzungsdauer, verbunden) kostenfrei. Bei mobilen Datennetzen wie UMTS oder LTE wird jedoch i.d.R. ein in einem Tarif inkludiertes Datenpaket verbraucht. Ist dieses Datenpaket aufgebraucht, ist nur noch eine langsame Verbindung möglich. Apps, welche zu viel Traffic in beschränkten Netzen verursachen, werden vom Anwender ggf. wieder schnell deinstalliert.

Daher sollte nicht nur generell der Datenverkehr minimiert werden, sondern bei notwendigem Transfer großer Datenmengen der Verbindungstyp beachtet werden: Eine App sollte größere Datenmengen nicht ohne explizite Zustimmung des Nutzers über mobile Datennetze herunterladen, dem Benutzer sollte die Möglichkeit geboten werden, den Download bis zur Verbindung mit einem WLAN zu verschieben.

Eine andere Option mobilen Traffic zu vermeiden ist es, dem Benutzer bei verfügbarem WLAN bewusst zu erlauben / anzubieten, später notwendige Daten herunterzuladen und lokal zu speichern (z. B. Kartenmaterial für einen Urlaubsort oder Vokabeln für eine neue Fremdsprache).

Aufgrund des Energieverbrauches sollten WLAN-Verbindungen, welche von der eigenen App geöffnet wurden, nach deren Gebrauch wieder geschlossen werden (nach Nutzerzustimmung), um den Energieverbrauch einzuschränken.

Mehrwert: Die aktive Verwaltung minimiert Traffic und Energieverbrauch.

Asset: Produkt

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: –

Beispiele: Nutze aktiv die Möglichkeiten zur aktiven Verwaltung der Netzwerkverbinden, die durch das entsprechende Device zur Verfügung gestellt werden:

- Nicht genutzte Datenverbindungen trennen
- Große Daten ausschließlich über WLAN downloaden

#110 Überwache den Speicherverbrauch.

Kenne den Bedarf der Applikation und warne den Anwender, wenn nicht genügend Speicher zur Verfügung steht, insbesondere auch vor der eigentlichen Installation (bzw. falls dies technisch nicht möglich ist beim ersten Start der App). Dabei sind die Möglichkeiten des internen und externen Speichers zu unterscheiden.

Wichtig ist auch beim Betrieb der Anwendung nicht mehr benötigten Speicher wieder freizugeben, d. h. Daten (seien es Dateien oder Datenbankeinträge) wieder zu löschen. Beispielsweise kann Kartenmaterial eines Urlaubszieles nach der Reise wieder gelöscht werden bzw. der Benutzer kann darüber informiert werden, dass die Daten schon länger nicht benötigt wurden. Im Anschluss wird er Benutzer gefragt, er die Daten löschen oder archivieren (nur noch in der Cloud bzw. auf einem PC speichern) möchte.

Mehrwert: Die Anwendung ist stabil und Installationen schlagen nicht ohne erkennbaren Grund fehl. Der Anwender hat die Möglichkeit für entsprechenden Speicherplatz zu sorgen, um die Anwendung im Anschluss erfolgreich zu installieren.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

Beispiele: Abfrage des verfügbaren Speichers vor der Installation und Bereitstellung von entsprechenden Anweisungen, um eine erfolgreiche Installation zu gewährleisten.

#111 Reduziere den Energieverbrauch auf ein Minimum.

Smartphones und mobile Anwendungen sind in unserem Alltag zum selbstverständlichen Begleiter geworden. Wie sehr man sich an all diese kleinen Helfer gewöhnt hat stellt man schnell fest, wenn man das Gerät einmal vergessen hat oder am Flughafen feststellen muss, dass das Smartphone aufgrund eines leeren Akkus nicht zur Verfügung steht. In dieser Situation fühlt sich der Anwender oft mehr als hilflos. Umso schlimmer ist es, wenn der Anwender feststellen muss, das der Grund für den leeren Akku eine bestimmte Anwendung ist. Diese Anwendung gefährdet die Möglichkeiten zur Kommunikation und Produktivität und würde mit hoher Wahrscheinlichkeit durch den Nutzer entfernt werden.

Mehrwert: Energieverschwender werden schnell erkannt und in den entsprechenden Foren als solche bekannt gegeben. Sie werden entweder erst gar nicht installiert oder werden nach Bekanntwerden von dem jeweiligen Gerät entfernt. Selbst eine herausragende Funktionalität ändert nichts an diesem Sachverhalt.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

- Testen Sie den Energieverbrauch mit und ohne ihre Anwendung.
- Identifizieren Sie energiehungrige Funktionen und minimieren Sie deren Energieverbrauch.

(Aktivieren Sie beispielsweise das GPS Modul nur, wenn es wirklich erforderlich ist)
- Führen Sie einen Energiebenchmark mit anderen Anwendungen durch.

4.3 Security

#112 Beschränke Zugriffsrechte auf das Nötigste.

Mobile Geräte bewahren heutzutage zahlreiche vertrauliche Daten auf, weshalb das Thema Datenschutz bei mobilen Anwendung stets relevant ist. Im Vergleich zu Desktop-Anwendungen sind mobile Anwendungen stark miteinander integriert, so dass bspw. häufig auf das zentrale Adressbuch oder den Kalender zugegriffen wird. Das Vertrauen in eine Anwendung ist daher eine zentrale Voraussetzung für ihre Akzeptanz, weshalb die erforderlichen Zugriffsrechte auf Daten und Hardware auf ein Minimum reduziert werden sollten. Die Anwendung sollte daher nur die Zugriffsrechte auf Daten und Hardware anfordern, die für das Funktionieren der Anwendung erforderlich sind. Zudem sollten Daten (auf die dann Zugriff besteht oder die erhoben werden [könnten]) nur für den Zweck der Anwendung genutzt und Übertragungen sowie zentrale Speicherungen der Daten unbedingt auf das notwendigste beschränkt werden.

Soweit möglich, sollten dem Anwender Wahlmöglichkeiten eröffnet werden. Beispielsweise könnte neben einer werbefinanzieren Version einer App (wobei die Werbung unter Nutzung zusätzlicher Daten selektiert wird) eine kostenpflichte werbefreie Version angeboten werden.

Durch einen bewussten und respektvollen Umgang mit den Daten des Anwenders wird Vertrauen geschaffen. Werden dagegen Missbräuche bekannt, werden Apps oft deinstalliert bzw. die Anzahl der Neuinstallationen sinkt aufgrund der negativen Berichterstattung.

Es ist jedoch auch zu beachten, dass Updates – sofern sie zusätzliche neue Rechte erfordern – nicht automatisch installiert werden können. Daher sollte man bei solchen Updates in der alten Version erklären, warum die neue Version zusätzliche Rechte benötigt (welche Funktionen damit realisiert werden), um einer steigenden Fragmentierung der eigenen Anwendung entgegenzuwirken.

Mehrwert: Das Vertrauen in eine Anwendung ist ein zentrales Kriterium für ihre Akzeptanz. Sofern Zweifel aufkommen, weil die Anwendung bspw. einen Zugriff auf Daten oder Hardware erfordert, deren Erfordernis nicht offensichtlich ist, droht sich die Akzeptanz einer Anwendung drastisch zu reduzieren. Mit der erforderlichen Transparenz und einem minimalen Zugriff auf sensible Daten bleiben das Vertrauen in die Anwendung und ihre Akzeptanz erhalten.

Asset: Produkt

Gültigkeit und Priorisierung: Ein »Soll« für jede Anwendung in jeder Domäne und auf jeder Plattform.

Referenzen: App Quality Alliance.

Beispiele: –

#113 Datenintegrität trotz mobiler Widrigkeiten sicherstellen.

Im Rahmen der Best Practice »Beschränke Zugriffsrechte auf das Nötigste« wird bereits die Vertraulichkeit von Daten betrachtet. Im Folgenden wird ergänzend dazu die Integrität von Daten fokussiert. App-Entwickler müssen Vorkehrungen treffen, um die Daten vor unberechtigter Manipulation (Korruption) und vor ungewolltem Verlust zu schützen. Bei mobilen Anwendungen gibt es spezifische Risiken, welche die Wahrscheinlichkeit einer Integritätsverletzung erhöhen sofern die App diese nicht hinreichend berücksichtigt. Hierzu zählt bspw. das Verlieren von Änderungen an einem Datum (Lost Update) durch Übertragungsfehler (App sendet Änderung, Server erhält diese jedoch aufgrund von Verbindungsproblemen

nicht). Auch durch weitere Ereignisse (App wird unterbrochen durch z. B. einen Telefonanruf oder durch Ausschalten des Telefons, leerer Akku) ist die Datenintegrität gefährdet. Zusätzlich darf das Risiko, die Daten durch Beschädigung oder Verlust des Gerätes zu verlieren, nicht vernachlässigt werden.

App-Entwickler sollten daher typische Szenarien testen und die Auswirkungen auf Datenintegrität betrachten. Anwender haben diesbezüglich hohe Ansprüche und im Schadenfall ist die Enttäuschung (und die damit verbundene PR) für den weiteren Erfolg einer App kritisch. Teilprobleme sind evtl. verteilte Datenspeicherung, Zugriff auf auf gemeinsame Datenbestände (z. B. Adressbuch, Terminkalender) durch verschiedene Apps und eingeschräkte Datenbanktechnologien.

Mehrwert: Verlust oder Korruption von persönlichen Daten zieht einen Vertrauensbruch nach sich und sorgt für negative Bewertungen.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

Testen Sie unter anderem folgende Szenarien:

- Verbindungsunterbrechung
- Schlechte Verbindung
- Wechsel der Verbindung (WLAN, UMTS, GPR)
- Entfernen der SD-Karte
- Schwache bzw. leere Batterie
- Unterbrechung der Stromversorgung
- Nicht genügend Speicherplatz
- Zusätzliche Ereignisse (Anruf, SMS, ...)

4.4 Interoperability

#114 Berücksichtige das Zusammenspiel mit anderen Anwendungen.

Mobile Geräte bieten viele Funktionen, die die Funktionalität Ihrer Anwendung negativ beeinflussen können. Eingehende Anrufe o. Ä. können die Nutzung von Apps beispielsweise unterbrechen. Genauso entstehen ggf. Beeinflussungen zwischen verschiedenen Applikationen, beispielsweise durch Popups oder Push-Nachrichten. Eine reibungslose Funktionalität auch bei Nutzung anderer beliebter Apps sollte daher unbedingt gegeben sein.

Mehrwert: Ihre Anwendung läuft sicher und stabil auf einem Smartphone und wird nicht negativ durch andere Anwendungen beeinflusst.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

Testen Sie folgende Szenarien:

- Eingehender/Ausgehender Anruf
- Senden/Empfangen einer SMS
- Nutzen des MP3 Players
- Nutzen der Kamera
- Nutzen von beliebten Anwendungen wie Facebook, Google+, WhatsApp

4.5 Portability

#115 Ermögliche eine reibungslose Installation auf unterschiedlichen Endgeräten.

Es gibt viele Möglichkeiten, um eine Anwendung auf einem mobilen Endgerät zu installieren, die zudem noch durch viele Einflussfaktoren, die der Anbieter nicht beeinflussen kann, beeinträchtigt werden können. Das Zusammenspiel all dieser Faktoren muss systematisch erfasst und entsprechend getestet werden. Also über welchen Anbieter (z. B. Shop), welches Medium (z. B. SD-Karte) oder welchen Weg (Wlan, UMTS, ...) soll die Anwendung auf das Gerät gelangen und wie sieht der Zustand des mobilen Endgerätes aus? Verfügbarer Hauptspeicher, Qualität der Datenverbindung, Verfügbarer Platz auf dem internen bzw. externen Speicher. Vor diesem Hintergrund sind ebenfalls Tests auf möglichst vielen relevanten Geräten und Betriebssystemversionen erforderlich. Ältere OS Versionen / Geräte sind häufig problematisch, der Markt sollte möglichst realistisch abgedeckt werden. Falls Abstriche gemacht werden, müssen diese deutlich kommuniziert werden (z. B. erst ab OS Version XY).

Mehrwert: Ihre Anwendung wird erfolgreich auf dem Endgerät installiert und somit ist die erste und wichtigste Hürde genommen, um den Anwender von Ihrer Lösung zu überzeugen. Es wird sichergestellt, dass die Anwendung auf unterschiedlichen Endgeräten und Systemen funktioniert.

Asset: Produkt

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

- Over the Air (OTA) Verhindern Sie einen Abbruch aufgrund einer schlechten Datenverbindung.
- Verhindern Sie einen Absturz aufgrund von einem Mangel an internem Speicher.
- Verhindern Sie einen Absturz aufgrund von einem Mangel an externem Speicher.
- Stellen Sie sicher, dass eine Installation über den Store fehlerfrei funtioniert.
- Portabilitätstest mit neuen Geräten muss regelmässig durchgeführt werden.
- Portabilitätstest einer neuen Version der Applikation muss regelmässig mit »älteren« Geräten durchgeführt werden.
- Check der wichtigsten / geschäftskritischen Funktionalitäten auf unterschiedlichen Versionen / Geräten.

5 Katalogasset Prozess

#201 Führe intensive Update-Tests durch.

Neue Updates können bei bestehenden Nutzern zu Problemen führen. Um nicht bereits gewonnene Kunden zu verlieren, sind daher Update-Tests ratsam. Vor allem Versionssprünge oder nicht weiter unterstützte Geräte und Betriebssystemversionen sind oftmals problematisch. Durch ältere Versionen sind außerdem bereits App-spezifische Inhalte auf dem Gerät gespeichert bzw. müssen überschrieben werden.

Mehrwert: Regelmäßige funktionierende Updates sorgen für ein positives Image und verbreitern die Kundenbasis. Das Update-Risiko wird minimiert.

Asset: Prozess

Gültigkeit und Priorisierung: »Muss« für alle Plattformen und Domänen.

Referenzen: Bezug zur Maintainability in ISO 25000.

Beispiele:

Führe Update-Tests mit vielen verschiedenen Szenarien durch:

- Art der Verbindung (WLAN, UMTS, ...)
- Quelle (Shop, SD Karte)
- Ziel (interner Speicher, externer Speicher)
- Versionsprung (beispielsweise von der ersten Version auf eine aktuelle Version)

#202 Liefere erforderliche Updates schnell.

Bei Problemen mit der aktuellen Version aber auch bei speziellen Anforderungen aufgrund neuer Betriebssystemversionen (z. B. iOS 7 Flat-Design) sollten Updates in kurzer Zeit ausgeliefert werden. Durch späte Updates auf aktuelle OS-Versionen bzw. Geräte können Probleme entstehen (bspw. grafische Probleme durch neue Bildschirmgröße). Dennoch sollten ausführliche Tests der Updates nicht vernachlässigt werden.

Mehrwert: Häufige Updates sprechen für ein gut gepflegtes Produkt und lassen zudem negative Bewertungen in der Historie des Shops verschwinden.

Asset: Prozess

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

- Versuchen Sie, Verbesserungsvorschläge zeitnah umzusetzen.
- Nehmen Sie Kritik den Wind aus dem Segel, indem Sie Probleme frühzeitig beheben.

#203 Teste mit Experten und Laien der Zielgruppe.

Neben systematisch strukturierten Tests (Testfälle) empfiehlt es sich, explorativ zu testen, um bisher nicht berücksichtigte Anwendungsfälle abzudecken und Probleme zu identifizieren. Hierbei finden normale Anwender durch unterschiedliches bzw. unbedarftes Vorgehen andere Probleme als Experten (zertifizierte Tester) und umgekehrt. Oft werden sowohl technische Probleme als auch Usability-Schwächen erst durch das Verhalten der jeweiligen Zielgruppe aufgedeckt, weshalb – falls möglich – auch die Endnutzer in das Testing einbezogen werden sollten. Testen Sie das reale Leben, indem Sie die Applikation alltäglich über einen längeren Zeitrahmen mit realen Geräten nutzen. Dies ist für Software allgemein und mobile Anwendungen im Speziellen ratsam. Im mobilen Bereich ist die Fehlertoleranz der Nutzer extrem gering

und bereits kleine Probleme führen oft zur Deinstallation sowie dem Wechsel zur Konkurrenz. Während eine Webseite schneller wieder besucht werden kann, ist die Schwelle für eine erneute Installation einer gelöschten App deutlich höher.

Dabei sollten auch Endanwender mit einbezogen werden. Dies sollte in jeder Phase – auch bereits mit den ersten Prototypen – erfolgen, um die Rückmeldungen möglichst früh in die weitere Entwicklung einfließen zu lassen. Die Endnutzer sollten so ausgewählt werden, dass sie verschiedene Nutzertypen der Zielgruppe repräsentieren. Dies bedeutet, dass zunächst die Zielgruppe analysiert und geclustert werden muss. Anschließend sind je Cluster Repräsentaten auszuwählen und entsprechende Testaufgaben vorzubereiten.

Mehrwert: Findung von Fehlern sowie Schwächen in der Benutzerführung der App, die bisher intern nicht aufgedeckt werden konnten, um so eine Deinstallation zu vermeiden. Die Komplexität von Kontext, Gerät und Applikation ist immens und kann nicht durch systematisches Testen allein abgedeckt werden. Erst das »wahre Leben« zeigt, ob die Applikation reif für den Markt ist.

Asset: Prozess

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: –

Beispiele:

- Tests mit Experten und Laien durchführen und Ergebnisse gegenüberstellen
- Einer Nutzergruppe verschiedene Anwendungsfälle (Use Cases) durchspielen lassen
- Bestimmte Fragestellungen durch die Zielgruppe beantworten lassen
- Erhöhte Geräte- und Betiebssystemabdeckung durch Nutzung der privaten Geräte der Tester.

#204 Überprüfe regelmäßig die Bewertungen in den Stores.

Mobile Anwendungen werden zum Großteil über die die Online Stores der großen Anbieter (Apple und Google) vertrieben. Sie sind somit die Anlaufstelle Nummer 1, nicht nur um die Applikation zu kaufen und zu installieren, sondern auch, um Informationen und Bewertungen zu bestimmten Applikationen zu sichten, bevor man sich für eine Lösung entscheidet. Bewertungskriterien sind hier sicherlich die Anzahl der Sterne, die eine Applikation erreicht, als Ranking aber auch die aktuellen Bewertungen auf der ersten Seite. So kann es durchaus passieren, das eine Applikation generell eine hohe Bewertung (viele Sterne) hat, aber auf der Startseite durchweg negative Bewertungen. Somit kann der potentielle Kunde zu dem Schluss kommen, das die aktuelle Version der Anwendung nicht fehlerfrei nutzbar ist. Diesem Umstand kann man nur entgegenwirken, indem man das Kundenfeedback ernst nimmt und eventuelle Probleme regelmäßig beseitigt. So kann man eine Vertrauenbasis weit in die Zukunft hinein aufbauen. Wenn ein Anwender sieht, das Fehler, die nach dem Kauf gefunden wurden, in der Regel schnell behoben werden, kann er sich leichter für eine Anwendung entscheiden.

Mehrwert: Aktive Teilnahme und kurzfristige Reaktionen auf negatives Feedback sorgen für ein positives Image. Viele Bewertungen enthalten auch Wünsche und Verbesserungsvorschläge, die Schwächen des Produktes aufzeigen.

Asset: Prozess

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: –

Beispiele: Besuchen sie regelmäßig die Stores von Google und Apple oder setzen Sie entsprechende Werkzeuge zur Analyse ein.

6 Katalogasset Organisation

#301 Virtualisiere Schnittstellen mit zentralisiertem Testinfrastruktur-management.

Unter Testumgebung versteht man die Menge von Software und Hardware, die nötig ist, um einen Test durchzuführen. Die Mächtigkeit einer Testumgebung variiert dabei über den Zeitraum, da ein Testobjekt zu Beginn der Entwicklung eine andere Testumgebung benötigt als die fertige Applikation. Die Komplexität von Testumgebungen korreliert dabei eng mit der Vernetztheit der Applikation. Letztere ist in mobilen Apps besonders hoch (teilweise über 30 Schnittstellen) und erfordert ein Umdenken bei Testumgebungen: Während der Entwicklung von Applikationen kann die Vielzahl von Schnittstellen nicht mehr ad-hoc und ungesteuert »nachprogrammiert« werden. Vielmehr muss eine zentral gesteuerte weitestgehend virtualisierte Testumgebung eingerichtet werden, da sich anders vielschichtige Schnittstellen wie GPS, Netzwerkprovider oder auch Temperatursensoren nicht mehr effizient entwickeln und testen lassen. Diese Testumgebung muss organisatorisch eindeutig zugeordnet werden können, da sie von vielen Rollen (z. B. Entwickler und Tester) genutzt wird.

Mehrwert: Virtualisierte Testumgebungen stellen sicher, dass ein effektives Shift-Left (oder Early-Error-Detection) möglich ist, obwohl die Abhängigkeiten zu einer Vielzahl variantenreicher Schnittstellen zunehmen. Virtualisierung hilft also, das Testen effizient zu gestalten und aufgrund der Einfachheit der Virtualisierung auch effektiv zu gestalten, indem mehrere Varianten unterschiedlicher Schnittstellen jeweils eigene Virtualisierungen für einen Test.

Asset: Organisation

Gültigkeit und Priorisierung: Gilt für alle Anwendungen auf allen Plattformen.

Referenzen: Die Notwendigkeit von Service-Virtualisierung in immer vernetzter werdenen Applikationen ist in unterschiedlichen Analysen belegt worden. Empirisches Zahlenmaterial findet sich z. B. in [Voke Research: Theresa Lanowitz: Service Virtualization, 2012].

Beispiele: –

#302 Treffe eine bewusste Technologieentscheidung.

Zunächst muss festgelegt werden für welche Plattformen (iOS, Android, Windows Phone, Firefox OS, ...) und welche Gerätetypen (Smartphone, Tablett) eine App zur Verfügung stehen soll. Erst nachdem dies geklärt ist kann in Abhängigkeit von der App-Idee entschieden werden mit welcher Strategie das Ziel erreicht werden kann. Folgende Strategien sind zu unterscheiden:

- Nativ: Die App wird für jede Plattform nativ entwickelt. Hierbei können die Besonderheiten der Plattformen und des jeweiligen SDK berücksichtigt werden um eine sehr gutes User Interface sowie eine hohe Performance zu erreichen. Die Zugriffsmöglichkeiten auf Gerätesensoren und -funktionen sowie Betriebssystemkomponenten (z. B. zentrales Adressbuch) sind hier am umfangreichsten.

- Web-App: Die App wird als Webseite entwickelt, welche dann vom Browser des jeweiligen Nutzers interpretiert wird. Der Zugriff auf einige Gerätesensoren mittels HTML5 & JavaScript wird von den meisten Browsern und Plattformen unterstützt. Das Layout kann mittels CSS und guter Frameworks ansprechend gestaltet werden. Offline-Funktionalität kann für sogenannte Client-Side-Web-Apps ebenfalls durch HTML5 erreicht werden. Die Grenzen des Ansatzes sind der Zugriff auf weitere Geräte- / Betriebssystems-Funktionen (z. B. Neigungssensor & Push-Nachrichten) sowie die schlechtere Performance und

die höhere Unsicherheit bzgl. der unterschiedlichen Interpretation der Seite durch die Browser.

- Hybride App: Hier wird eine WebApp mit einer nativen Anwendung kombiniert. Die WebApp wird weiterhin vom Browser (auch wenn dieser nicht erkennbar ist) interpretiert, der native Rahmen ermöglicht den Zugriff auf weitere Gerätefunktionen (durch bereitstellen einer entsprechenden JavaScript API) und die Installation der Anwendung. Es gibt eine Vielzahl von Frameworks welche den nativen Rahmen bereitstellen und bei der Entwicklung der App unterstützen.

Neben diesen gängigen Ansätzen gibt es folgende – ggf. für bestimmte Apps passenderen – Ansätze:

- Apps mit eigenem Interpreter: Wie im Falle der hybriden App wird die App mit weit verbreiteten Technologien (i.d.R. auch HTML5, JavaScript und CSS) erstellt. Es wird jedoch nicht der Browser als Interpreter verwendet, sondern der native Rahmen bringt einen eigenen Interpreter mit. Diese speziellen Interpreter wurden bzgl. Performance für solche Apps verbessert, sodass darin der Hauptvorteil gegenüber normalen hybriden Apps liegt.

- Cross-Compiled: Die App wird in einer Sprache entwickelt und ein Cross-Compiler erzeugt native Apps für verschiedene Plattformen. I.d.R. müssen zusätzliche Anpassungen z.B. für die GUI vorgenommen werden.

- Model-Generated: Aus einem Modell wird für verschiedene Plattformen Quellcode für die spätere Anwendung automatisch erzeugt. Manuelle Anpassungen sind dann bspw. noch für die GUI nötig. Die heutige Praxisrelevanz des Ansatzes liegt in der Generierung von einzelnen Komponenten von Anwendungen. Es wird jedoch eine zunehmende Mächtigkeit der Systeme erwartet.

Die jeweilige Entscheidung muss davon abhängig gemacht werden, welche Ziele mit der Anwendung verfolgt werden. Hierbei gilt es meist, den bestmöglichen Kompromiss zwischen der Wartbarkeit der Anwendung, sowie der Usability zu finden. Gleichzeitig spielen auch die zur Verfügung stehenden Funktionalitäten eine zentrale Rolle. Während in der Vergangenheit die Nutzung geräteabhängiger Funktionen wie bspw. des GPS-Moduls nativen Anwendungen vorbehalten war, finden heutzutage zunehmend Webstandards Einzug, die ebenfalls diese Funktionen unterstützen. Native Anwendungen hingegen können – bei Bereitstellungen einzelner Anwendungen für verschiedene Plattformen – insgesamt wartungsintensiver sein, da sie im Vergleich zu einer Webanwendung nicht zentral zur Verfügung gestellt werden, dahingegen wird bei hohen Anforderungen an die Usability meist eine native Anwendung bevorzugt. Es gilt daher, den geeigneten Trade-Off zwischen diesen verschiedenen Punkten zu finden.

Mehrwert: Die Anwendung bietet den bestmöglichen Trade-Off zwischen den Vor- und Nachteilen der verschiedenen Technologien bezogen auf die jeweiligen Anforderungen. Zwischen Wartungskosten und den Vorzügen für den Anwender wurde daher bestmöglich abgewogen. Es handelt sich hierbei um eine strategische Entscheidung, die langfristige Auswirkungen hat.

Asset: Organisation

Gültigkeit und Priorisierung: Ein »Muss« für jede Anwendung in jeder Domäne und auf jeder Plattform.

Referenzen: SIGS Datacom / FZI Poster.

Beispiele:

- Eine hohe Latenz spricht spricht beispielsweise gegen HTML 5 Applikationen.
- Hohe Sicherheitsanforderungen sprechen dagegen eher für eine HTML 5 Applikation.

#303 Teste nicht nur die App.

Die Testaufgaben beziehen sich entlang des modernen Testverständnisses (vgl. [ISTQB]) wenigstens auf alle (Zwischen-)-Ergebnisse der Software-Entwicklung sowie die für die Nutzung notwendigen externen Komponenten. Der Begriff Testobjekt bezieht sich dabei auf genau die Artefakte, die getestet werden müssen, um die fertige Anwendung möglichst risikolos betreiben zu können. Die Menge relevanter Testobjekt variiert nun deutlich von denen klassischer Desktop-Anwendungen. Im App-Bereich muss z. B. der Marketplace für eine App ebenfalls getestet werden, da ein Nicht-Funktionieren dieser Komponente die Gesamtanwendung gefährdet. Ebenfalls muss z. B. das monetäre Modell einer App (womit möchte man wie Geld verdienen) getestet werden, da anderen-falls die Nichtzulassung im Marketplace droht (z. B. bei fehlender Compliance zu Apple). Des weiteren muss auch das Netz in seinen unterschiedlichen Ausprägungen getestet werden (GSM, EDGE, UMTS, LTE), da es je nach Nutzung durch die Applikation unterschiedlich relevant für dessen Funktionieren sein kann. Diese Vielschichtig-keit der neuen Testobjekte muss sowohl im Testkonzept als auch im Testprozess entsprechend berücksichtigt werden.

Dabei sind auch unterschiedliche Geräte mit unterschiedlichen OS-Versionen (verursacht durch das Fragmentierungsproblem) einzubeziehen. Um dennoch mit beschränkten Ressourcen testen zu können, sollten Geräteklassen gebildet werden (z. B. nach OS, Bildschirmgröße, Arbeitsspeicher und Prozessor) und je Klasse ein repräsentatives Gerät in den Test aufgenommen werden.

Mehrwert: Für den Endanwender gibt es pro Applikation nur eine gefühlte Qualität. Auftretende Probleme werden nur selten einer Ursachenanalyse zugeführt, um den tatsächlichen Grund zu erheben. Von daher entscheidet meist das schwächste Glied der gesamten Produktionskette über die Gesamtqualität. So wird die beste Applikation scheitern, wenn sie nicht durch den Marketplace entsprechend installiert werden kann. Und auch wenn sie funktioniert können z. B. Probleme mit schwachen Netzverbindungen (z. B. Edge) die Gesamtwahrnehmung signifikant beeinflussen.

Asset: Organisation

Gültigkeit und Priorisierung: »Muss« für alle Anwendungen auf allen Plattformen.

Referenzen: Das holistische Qualitätsverständnis ist im Testbereich als auch im Controlling-Bereich mittlerweile etabliert (vgl. z. B. [Simon&Simon: Ganzheitliches Qualitätsmanagement, 2012]). Die Vielschichtigkeit von Testobjekten wird insbesondere im ISTQB-Syllabus hervorgehoben.

Beispiele: –

#304 Teste und entwickle vieles nur einmal: Multi-Channel-Entwicklung und Test.

Anders als die herkömmliche Entwicklung von Desktop-Applikationen muss die Entwicklung mobiler Anwendungen eine Vielzahl unterschiedlicher Zielplattformen berücksichtigen. Einige diese Vielfalt begründenden Dimensionen sind:

- Unterschiedliche Betriebssysteme (z. B. iOs, Android, Windows)
- Unterschiedlichen Betriebsystem-Versionen (z. B. von Android 2.1. bis zum neuen Android 4.3)
- Unterschiedliche Brandings (z. B. Vodafone-Branding, T-Mobile Branding)
- Unterschiedliche Hardware (z. B. Nokia, Samsung, HTC)
- Unterschiedliche Netzwerke (z. B. GSM, edge, UMTS, LTE in unterschiedlichen Regionen)
- Unterschiedliche Deployment-Möglichkeiten (z. B. über zentrale Marketplaces oder dezentrale Bluetooth-Versendung)

Die Zielvielfalt darf im Entwicklungs- und Testprozess nicht durch ein bloßes mehrfaches Instanziieren eines Ein-Ziel-Prozesses abgebildet werden, sondern muss durch einen entsprechenden Varianten-unterstützenden Prozess realisiert werden. Dabei muss jeder Schritt, jede Aktivität und jedes (Zwischen-) Ergebnis daraufhin

überprüft werden, was Gemeinsamkeiten zwischen allen betrachteten Zielgeräten sind, und welche Spezifika es pro Zielsystem gibt. Natürlich sind auch Mischformen denkbar, bei denen ein Spezifikum für einen Teil der Zielgeräte relevant sind.

Mehrwert: Trotz der hohen Variantenvielfalt bleibt der Aufwand für das Testen und Entwickeln beherrschbar.

Asset: Organisation

Gültigkeit und Priorisierung: Dieses Pattern sollte für alle Projekte eingesetzt werden, die nicht nur eine mobile Zielplattform fokussieren. In solchen Fällen ist die Empfehlung allerdings universell gültig, um effizient mobile Anwendungen zu erstellen. Da es aber lediglich den Aufwand reduziert und die Effektivität nicht direkt erhöht, ist es keine zwingende Notwendigkeit, wohl aber eine dringende wirtschaftliche Empfehlung.

Referenzen: Als Referenzen ist hier der Wissenschaftsbereich der »Variantenreichen Software-Entwicklung« anzugeben. IBM hat hierzu den Bereich des Multi-Channel-Development kreirt. Das German Testing Board (GTB) hat hierzu einen Beitrag mit dem Titel Multi-Channel-Testing veröffentlicht (Literaturverweise kommen anschließend).

Beispiele: Einige Beispiele für derartige Faktorisierungen:

- In der Anforderungsermittlung (Requirements-Engineering) sind ein Großteil der Anforderungen für alle Zielsysteme identisch. Diese gilt es explizit als solche zu markieren und hoch zu werten.
- Die High-Level-Testfallermittlung folgt diesen universellen Anforderungen nach und ist ebenfalls nur einmal durchzuführen.
- Testumgebungen (z.B. zu Facebook) sind ebenfalls daraufhin zu überprüfen, ob die Schnittstellen nicht für alle Zielsysteme identisch sind und damit für alle Testumgebungen identisch angeboten werden können.

#305 Kopiere nicht bereits vorhandene Lösungen 1 zu 1 auf mobile Geräte.

Die Erwartungshaltung mobiler Endanwender unterscheidet sich von denen der Desktop-Anwendern.
Es sollte daher gut überlegt sein, ob eine bereits als Desktop- oder klassische Web-Anwendung verfügbaren Inhalte und Aufbereitungen von Daten in ähnlicher Form mobil verfügbar gemacht werden (z.B. in Form einer Responsive Website) oder ob eine dedizierte mobile Anwendung entwickelt wird. Aufgrund der unterschiedlichen Anforderungen ist in den meisten Fällen eine dedizierte App zu empfehlen. Aktuell zeichnet sich ein Trend hin zu Apps sogar auf dem Desktop ab (Windows 8).

Mehrwert: Die Anwendung richtet sich nach den Anforderungen mobiler Endanwender. Sie stellt Inhalte entsprechend dar und ist somit auch durch den Anwender bedienbar, was eine Grundvoraussetzung für eine Nutzung der Anwendung und somit ihrer Akzeptanz darstellt.

Asset: Organisation

Gültigkeit und Priorisierung: »Soll« für alle Plattformen und Domänen.

Referenzen: App Quality Alliance

Beispiele: Einige Beispiele für derartige Faktorisierungen:

- Auf einem kleinen Bildschirm muss man öfter auf einen Wizard zugreifen, als auf einem großen Bildschirm.
- Abstände zwischen Interaktionselementen müssen größer sein, damit sie mit dem Finger bedienbar sind.
- Inhalte sollten nur bei Bedarf sichtbar werden, z.B. durch ausklappbare Menüs / Inhalte.

Der BITKOM vertritt mehr als 2.100 Unternehmen, davon rund 1.300 Direktmitglieder mit 140 Milliarden Euro Umsatz und 700.000 Beschäftigten. 900 Mittelständler, mehr als 100 Start-ups und nahezu alle Global Player werden durch BITKOM repräsentiert. Hierzu zählen Anbieter von Software & IT-Services, Telekommunikations- und Internetdiensten, Hersteller von Hardware und Consumer Electronics sowie Unternehmen der digitalen Medien und der Netzwirtschaft.

BITKOM

Bundesverband Informationswirtschaft,
Telekommunikation und neue Medien e.V.

Albrechtstraße 10 A
10117 Berlin-Mitte
Tel.: 030.27576-0
Fax: 030.27576-400
bitkom@bitkom.org
www.bitkom.org