infoteam
software

WHITEPAPER

Medical Apps

Medical Apps

Mobile-App-Entwicklung im regulierten Medizinprodukteumfeld – ein Widerspruch?

1 Einführung und Motivation

Die weite Verbreitung von leistungsfähigen mobilen Endgeräten wie Smartphones und Tablets über die unterschiedlichsten Nutzergruppen hinweg hat nicht nur bei Verbrauchern zu einer Explosion an mobilen Applikationen – kurz Apps – geführt. Nahezu für jede Lebenssituation gibt es diese kleinen nützlichen Programme, deren Erfolgsgeheimnis ihre einfache Handhabung und intuitive Bedienung sind.

Auch etwa 80.000 medizinische Apps sind über die Stores von Google, Apple und Co. verfügbar. Das Spektrum reicht von der Fitness- und Ernährungs-App über Dosierungsrechner für Medikamente bis hin zu Spezial-Apps für die diagnostische Darstellung von Ultraschallbildern oder die Patientenüberwachung. Viele dieser Apps sind Medizinprodukte und unterliegen somit regulatorischen Auflagen. In Europa benötigen sie folglich ein CE-Zeichen und in den USA eine Zulassung der Food and Drug Administration (FDA). Beides erfordert die Einhaltung von Gesetzen, Normen und Richtlinien und erhöht dadurch letztlich den Aufwand für die Entwicklung und Inverkehrbringung.

Für die Entwicklung von medizinischen Apps gelten andere Randbedingungen als z. B. für eine Routenplaner-App: Wie behandle ich als Inverkehrbringer – der Funktionalität der App angemessen – Fragen zu Dokumentation, Wartung und Risikomanagement für die unterschiedlichen Zielplattformen?

Rund um das Thema App-Entwicklung ist inzwischen eine Vielzahl an ausgereiften Tool-Ketten entstanden – von der Entwicklung über Cross-Platform-Ansätze bis hin zum automatisierten Test auf dem Smartphone. Dieses Whitepaper gibt einen Überblick über die gängigen technischen Möglichkeiten zur Realisierung von Medical Apps und beschreibt die regulatorischen Randbedingungen, die es zu beachten gilt.

2 Überblick über die Programmierung von mobilen Applikationen

2.1 Native Apps

Native Apps im engeren Sinn zeichnen sich dadurch aus, dass sie speziell an die Zielplattform angepasst sind und nur in den herstellereigenen Stores für die Installation zur Verfügung stehen.

Für die Entwicklung dieser Apps stellen die Hersteller der Betriebssysteme eigene Tools zur Verfügung. Diese Tools sind kostenlos und optimieren die Performance der erstellten Apps für die jeweilige Zielplattform. Zudem stehen sämtliche vom Hersteller angebotenen Schnittstellen der Geräte zur Verfügung, z. B. zu einem Beschleunigungssensor, GPS oder Bluetooth. Allerdings sind die nativen Entwicklungsumgebungen der verschiedenen Hersteller untereinander inkompatibel. Das hat zur Folge, dass die Entwicklung einer App für die drei verbreitetsten Betriebssysteme Android, iOS und Windows Phone auch dreifachen Implementierungsaufwand bedeutet.

2.1.1 Android

Android hat von Apple die Marktführerschaft in Bezug auf die verkauften Geräte übernommen. Über Google Play werden native Apps für Android verbreitet. Basis der Android-App-Entwicklung ist Java als Programmiersprache und das Android Software Development Kit (SDK). Als Entwicklungsumgebung stehen dem Entwickler Eclipse, das das SDK über ein Plug-in einbindet, oder die neue Entwicklungsumgebung Android Studio zur Verfügung.

Das Google-Betriebssystem hat eine sehr hohe Verbreitung im Markt. Der enorme Variantenreichtum der Geräte mit unterschiedlichen Formfaktoren, Auflösungen und Versionen des Betriebssystems relativiert diese Einschätzung aber wieder.

2.1.2 Apple iOS

Obwohl iOS im Vergleich zu Android einen niedrigeren Marktanteil hat, sind Apples Umsätze mit Apps höher. Daher bleiben Apples iPhone und iPad nach wie vor sehr wichtige Plattformen für Apps.

Für die Entwicklung nativer iOS-Apps bietet Apple die kostenlose IDE XCode an, die allerdings nur auf Apple-Hardware läuft. Programmiersprache ist klassischerweise ObjectiveC, seit 2014 Swift als modernere Alternative.

Ab iOS 8 bietet Apple mit dem HealthKit eine einheitliche Schnittstelle für Fitness- und Gesundheits-Apps an, woran Hersteller andocken können. Zudem ist die Anzahl der Varianten von iPhones vergleichsweise überschaubar. Die Validierung ist bei Medical Apps für iOS somit besser zu lösen.

2.1.3 Windows Phone

Die Verbreitung des Windows Phone ist seit Jahren langsam im Aufwind und soll Prognosen zufolge insbesondere in Europa bald der Verbreitung von iOS nahekommen. Dennoch ist es in manchen Märkten praktisch nicht existent.

Microsoft bietet als native Entwicklungsumgebung für Windows Phones Visual Studio an. Die verwendete Programmiersprache ist C#.

2.1.4 Weitere Betriebssysteme

Die meisten App-Hersteller konzentrieren sich auf iOS und Android. Weniger verbreitete oder aktuelle Betriebssysteme wie Bada, Symbian, Firefox OS oder Blackberry werden kaum oder gar nicht unterstützt.

2.1.5 Cross-Compilation

Möchte man native Apps für mehrere Zielplattformen herstellen, so entwickelt, testet, validiert und wartet man gleich mehrfach – mindestens einmal pro Plattform. Insbesondere für Medical Apps bedeutet das einen ungleich höheren Aufwand.

Eine Alternative hierzu ist die Nutzung von Cross-Compilation Tools. Im Gegensatz zu Cross-Platform Tools ist das Ergebnis eine komplett native App ohne HTML5- oder JavaScript-Anteile. Die Performance und die unterstützten Schnittstellen sind mit klassischen nativen Apps praktisch identisch. Allerdings ist die Anzahl unterstützter Plattformen übersichtlich.

Dieser Ansatz ist oft dann die beste Wahl, wenn schon existenter funktionierender Code wiederverwendet werden soll. So kann man mit dem auf dem Mono-Projekt basierenden Xamarin bereits existierenden C#-Code verwenden (siehe Abb. 1). Bei QT Mobile ist C++ die Sprache der Wahl. Die jeweiligen Cross-Compilation Frameworks stellen viele Basisbibliotheken zur Verfügung.

Eine der großen Herausforderungen bei Cross-Compilation sind die unterschiedlichen Nutzererfahrungen und GUI-Konzepte. Während iPhones nur den Home-Button besitzen, haben Android-Handys mehrere Hardware-Buttons. Auch die GUI-Controls unterscheiden sich sehr stark, teilweise sogar von Version zu Version des gleichen Betriebssystems.

Xamarin setzt daher auf die Integration der vom jeweiligen Hersteller gegebenen GUI-Technologien und Editoren. Aufgrund dessen ist die GUI-Anbindung für jede Plattform einzeln zu programmieren, was problemlos umzusetzen ist, doch wird streng genommen nur die Programmlogik cross-kompiliert.

QT Mobile dagegen bietet noch keine eingebaute Möglichkeit, die Apps „nativ“ aussehen zu lassen. Der Hersteller kann ein beliebiges Look-and-feel aber selbst stylen.

Cross-Compilation Tools sind nicht kostenlos, stellen jedoch eine gute Variante dar, die Entwicklung von Apps für verschiedene Plattformen so einfach wie möglich zu gestalten und dabei die Vorteile nativer Apps beizubehalten, wenn man bereit ist, sich mit den unterschiedlichen UI-Technologien auseinanderzusetzen, oder einen einheitlichen Look für alle Plattformen akzeptiert.

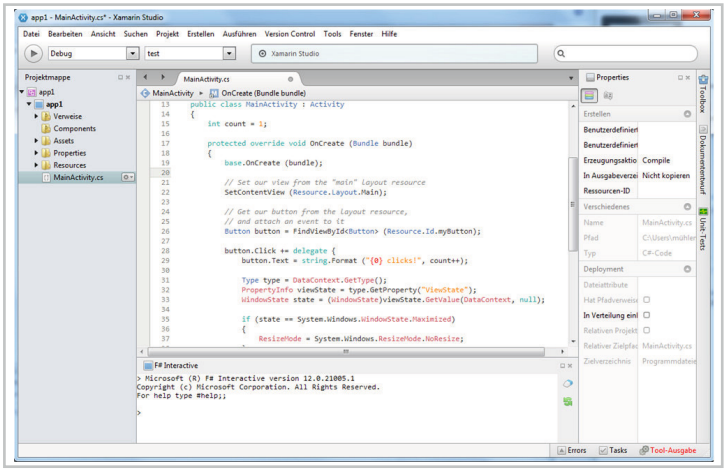

Abbildung 1: Cross-Compilation Framework Xamarin

2.2 Web Apps und hybride App

2.2.1 HTML5 im Aufwind

Web Apps sind Webseiten, die meist auf HTML5 und CSS3 basieren und für die Darstellung und Bedienung auf dem Smartphone optimiert sind. HTML5 hat als universelle Oberflächensprache den großen Vorteil, dass es auf praktisch allen Geräten dargestellt werden kann.

Die Programmlogik wird meist in JavaScript bereitgestellt. JavaScript wurde ursprünglich als nicht typsichere, Client-seitige Scriptsprache für dynamisches HTML in Webbrowsern entwickelt. Inzwischen existieren zahlreiche ausgereifte Ansätze, in anderen Sprachen typsicher zu programmieren und dann daraus JavaScript-Code zu generieren. Das Google Web Toolkit (GWT) mit Java als Ausgangssprache ist ein weitverbreitetes Beispiel dafür.

Der größte Nachteil dieses Programmieransatzes ist die Interpretation der JavaScript-Logik, deren Ausführung im Vergleich zu nativen Apps langsamer erfolgt. Dieser Nachteil ist durch Just-in-time- (JIT)Compilation allerdings deutlich vermindert worden, sodass der Unterschied für viele Anwendungen keine spürbaren Folgen hat.

Web Apps können die Distribution über die Stores umgehen, da sie nur als reine Webseite zur Verfügung gestellt und im Browser des Smartphones angezeigt werden. Sie benötigen allerdings – anders als native Apps – unbedingt eine Server-Infrastruktur.

2.2.2 Hybride Apps

Ist eine Verbreitung über die Stores oder die Nutzung der Geräteschnittstellen gewünscht, bieten sich hybride Apps an. Das sind native Apps, die jedoch über ein Browser-Control HTML5-Inhalte ähnlich einer Web App präsentieren (siehe Abb. 2). Diese Apps können phasenweise oder permanent ohne Webserver auskommen.

Zahlreiche Tools der unterschiedlichen Preisklassen – von kostenlos bis hochpreisig – unterstützen diesen Cross-Platform-Ansatz.

Eine Einschränkung der hybriden Apps macht bislang Apple: Die bereits erwähnte JIT-Compilation wird (im Gegensatz zu Web Apps in Safari) erst ab iOS 8 für diese Art von Apps unterstützt. Die Folge: Cross-Plattform Apps sind auf den bisherigen iOS-Geräten deutlich langsamer, während die JIT-Unterstützung auf Android-Geräten schon länger durchgehend gegeben ist.

2.2.3 Unterstützende Frameworks

Auch für Web Apps und hybride Apps sind Bedienkonzepte und Styling relevant. Die Frage, wie die App sich an unterschiedliche Bildschirmgrößen anpasst und für jedes Betriebssystem ein entsprechendes natives Look-and-feel erzeugt, ist dabei von entscheidender Bedeutung. Bei Web Apps und hybriden Apps kann dies mithilfe von JavaScript-Bibliotheken und Frameworks erreicht werden.

Das ursprünglich von Twitter entwickelte Bootstrap-Framework erlaubt es, Webseiten und HTML5-Apps mit der Bildschirm- auflösung skalieren zu lassen: Während die Webseite auf einem Full-HD-Monitor sehr detailliert erscheint, werden auf einem Tablet bereits erste Informationen vereinfacht, auf einem Smartphone schließlich nicht unbedingt notwendige Informationen ausgeblendet und der Rest automatisch untereinander statt nebeneinander angezeigt.

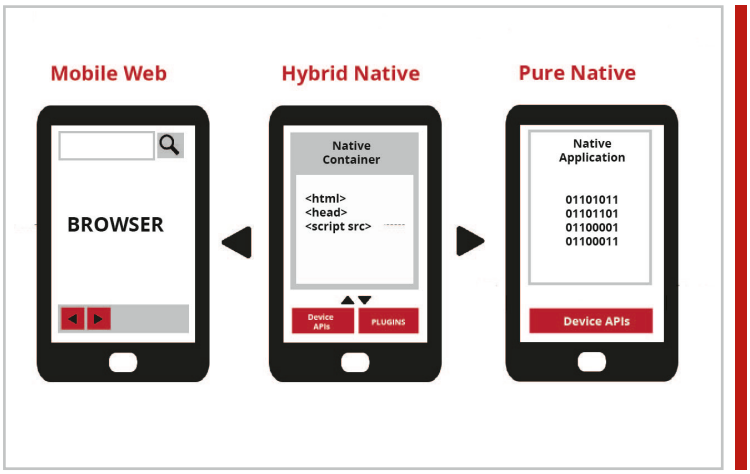

Abbildung 2: Darstellung der unterschiedlichen Typen von Apps

Für das native Look-and-feel existieren ebenfalls Bibliotheken, die es erlauben, die HTML5-Oberflächenelemente je nach Zielplattform annähernd genauso aussehen zu lassen wie eine native App auf dem jeweiligen System. Ein Beispiel hierfür ist das JQuery Mobile Framework.

Fast alle Cross-Platform Frameworks basieren auf dem Open Source Framework Apache Cordova. Das bekannteste Beispiel hierfür ist Adobe Phonegap. Cordova wandelt die JavaScript- und HTML-Inhalte in eine App um, die in den Stores zur Verfügung gestellt werden kann. Zudem liefert Cordova JavaScript-Schnittstellen für die wichtigsten Gerätefunktionen wie GPS und Beschleunigungssensor, die somit in Cross-Platform Apps verwendet werden können. Allerdings sind nicht alle Schnittstellen verfügbar, die die nativen Apps zur Wahl stehen.

Verschiedene Cross-Platform-Ansätze bie-ten sich als Alternative zur klassischen nativen App-Entwicklung an. Mit ihnen lässt sich der Aufwand für die Implementierung und Pflege reduzieren. Dabei ist zwischen HTML-basierten hybriden-Apps und cross-kompilierten Apps zu unterscheiden. Eine Übersicht über die verschiedenen Deployment-Szenarien für Apps gibt Abbildung 3.

Zu beachten ist insbesondere, dass bei Medizinprodukten im Normalfall nicht die Implementierung, sondern die Dokumentation und die Durchführung der normenkonformen entwicklungsbegleitenden Prozesse (Risikomanagement) den größeren Anteil am Aufwand haben. Auch sie werden aber durch die Wahl der Technologie beeinflusst.

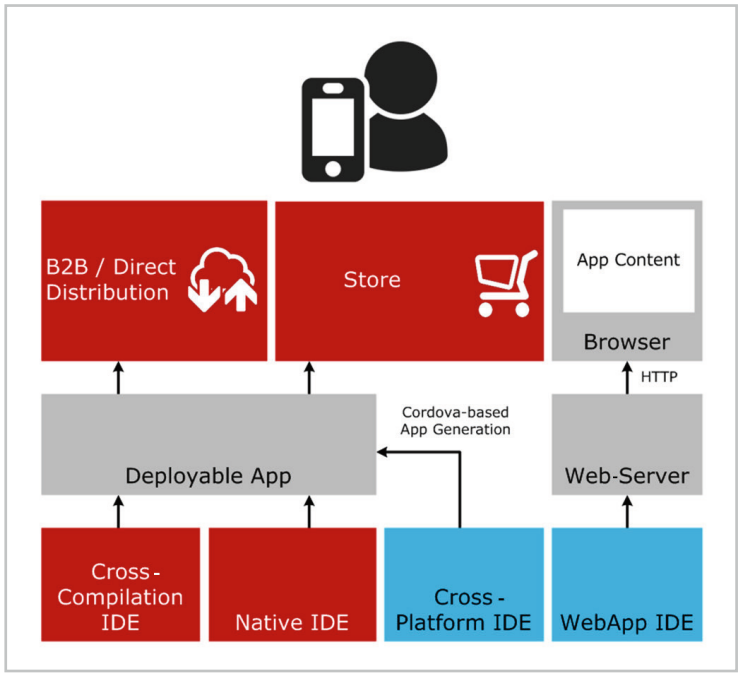

Abbildung 3: Übersicht der verschiedenen Deployment-Szenarien für Apps

3 Wann ist eine App ein Medizinprodukt?

Es gibt inzwischen eine Vielzahl von Apps mit medizinischem Kontext in den unterschiedlichen Stores. Doch sind diese Apps Medizinprodukte? Zu Medizinprodukten zählen Geräte und auch Software, die ein Hersteller für die Diagnose, Therapie, Verhütung oder Linderung von Krankheiten, Verletzungen oder Behinderungen des Menschen (in den USA auch von Tieren) vorgesehen hat. Nicht darunter fallen z. B. die meisten Fitness-Apps sowie Apps zur Ernährungsberatung.

In seiner Zweckbestimmung kann der Hersteller die Verwendung als Medizinprodukt auch verbieten, wenn diese naheliegt, aber nicht beabsichtigt ist. Dann allerdings dürfen Handbuch, Werbematerialien oder im Fall von Apps die Beschreibung im Store auch nicht auf diagnostische oder therapeutische Funktionen im Sinne eines Medizinproduktes hinweisen. Im Zweifelsfall muss der vorherzusehende Missbrauch – nämlich der Einsatz entgegen der Zweckbestimmung – wirksam verhindert werden.

Insbesondere bei Apps existiert in der Anwendung eine Grauzone, obwohl dieselben Regularien und Normen gelten wie für jede andere Software auch. Um Medizinprodukte – und damit auch Mobile Medical Apps - in Verkehr bringen zu dürfen, muss ein Hersteller deren Konformität mit den harmonisierten Normen nachweisen können.

3.1 Qualitätsmanagement und Entwicklungsprozess

Der Hersteller von aktiven Medizinprodukten in Gestalt von Apps muss ein Qualitätsmanagementsystem nach der Norm EN ISO 13485 vorweisen. Der Softwareentwicklungsprozess muss der Norm IEC 62304 folgen, womit auch die daraus resultierenden Traceability-Anforderungen zu erfüllen sind.

Stellt der Hersteller spezielle Hardware-Erweiterungen wie ein anschließbares Blutdruckmessgerät für iPhone und Co. bereit, so wird er mit Anforderungen an das Gesamtgerät wie z. B. der Betrachtung der elektromagnetischen Verträglichkeit (EMV) und der elektrischen Sicherheit konfrontiert, die sich aus der IEC 60601 ergeben. Handelt es sich dagegen um eine Standalone-App, kommt zukünftig die DIN EN 82304 zum Tragen.

3.2 Risikomanagement und Usability Engineering

Der Hersteller muss auf Basis der DIN EN ISO 14971 ein vollständiges Risikomanagement betreiben, das die Risiken für Patienten, Anwender und Dritte von Beginn an erfasst, behandelt und abschließend Risiken und Nutzen zueinander ins Verhältnis setzt.

Dies schließt auch die medizinisch fundierte Bewertung möglicher Schäden ein. Eng verzahnt mit dem Risikomanagement ist das ebenfalls geforderte Usability Engineering durchzuführen, in Europa unter Beachtung der IEC 62366. Fehler in der Gebrauchstauglichkeit führen insbesondere bei medizinischer Software zu oft folgenschweren Fehlbedienungen.

Daher ist im Usability Engineering neben dem medizinischen Zweck auch die Benutzergruppe mit ihren Fähigkeiten und Einschränkungen genau zu umreißen. Die Hauptbedienfunktionen und Anforderungen an die Usability sind zu spezifizieren und schließlich zu validieren, um möglichen Bedienfehlern vorzubeugen. Mockups unterstützen die Validierung von Usability-Konzepten (siehe Abb. 4)

Eine Herausforderung bei der App-Entwicklung ist die unkontrollierbare Verbreitung über die Stores. Wenn der Hersteller die Nutzergruppe nicht einschränkt, wie kann er dann sicherstellen, dass sich nur Ärzte und nicht auch Patienten diese App installieren und sich selbst medikamentieren?

Abbildung 4: Mockups unterstützen die Validierung von Usability-Konzepten

Hierfür gibt es verschiedene technische Lösungen, von alternativen Vertriebswegen bis hin zu Zertifizierungsmechanismen oder einer zentralen Nutzerregistrierung.

3.3 Zulassungsverfahren in Europa

Das europäische Zulassungsverfahren ist abhängig von der Klasse des Medizinproduktes, das nach dem Anhang IX der MDD bestimmt wird, und fordert neben der Einhaltung der genannten harmonisierten Normen auch eine klinische Bewertung.

In Europa endet die Zulassung meist mit einer Konformitätserklärung des Herstellers, der jedoch – außer bei unkritischen Produkten der Klasse I – von einer benannten Stelle auditiert sein muss. Das gilt auch für Medical Apps, wenn sie ein Medizinprodukt sind.

Um die Konformität seiner App mit den grundlegenden Anforderungen der MDD nachzuweisen, muss der Hersteller u. a. folgende Dokumente erstellen:

- IEC 62366 konforme Gebrauchstauglichkeitsakte
- DIN EN ISO 14971 konforme Risikomanagementakte
- IEC 62304 konforme „Software-Akte“

In manchen Spezialfällen können Handy-Apps die sehr spezifischen Anforderungen für eine Zulassung nicht erfüllen. Eine bildbasierte diagnostische Befundung beispielsweise ist in vielen Fällen nicht als Handy-App zulässig, da schlicht das Display zu klein oder die Auflösung zu gering ist. Der ZVEI-Leitfaden „Mobile Geräte und Apps in der Medizin“ listet viele dieser Ausschlussgründe und ihren normativen Ursprung auf.

3.4 Betreiberpflichten durch Server-Logik

Die beschriebenen Pflichten sind die des Inverkehrbringers. Bei Webseiten und Web Apps, aber auch bei Webservices mit Medizinproduktcharakter kann das Hosten dieser Dienste als Betrieb eines Medizinproduktes beurteilt werden.

Findet z. B. bei einem Dosierungsrechner die eigentliche Berechnung nicht in der App, sondern auf einem Server des Herstellers statt, muss dieser unter Umständen die Pflichten eines Medizinproduktebetreibers nach der Medizinproduktebetreiberverordnung erfüllen, z. B. ein Medizinproduktebuch führen und weitere Normen wie die DIN EN 80001 beachten.

3.5 Zulassung in den USA

In den USA genügt eine Konformitätserklärung nicht, vielmehr wird die Zulassung durch die Food and Drug Administration (FDA) erteilt. Die FDA erkennt einige der harmonisierten Normen an (z. B. die DIN EN ISO 14971 für das Risikomanagement), hat zu anderen Themen wie dem Usability Engineering und dem Softwareentwicklungsprozess jedoch weitere, teils darüber hinausgehende Anforderungen. Insbesondere muss das folgende Bedingungswerk der FDA berücksichtigt werden:

- Quality System Regulations (21 CFR part 82)
- Electronic records and electronic signatures (CFR 21 part 11)
- Guidance Document „Principles of Software Validation“
- Guidance Document „Contents of Premarket Submissions for Software Contained in Medical Devices”
- Guidance Document „Applying Human Factors and Usability Engineering to Optimize Medical Device Design”

Die FDA recherchiert selbstständig auch in den unterschiedlichen Stores und macht Apps ausfindig, die ohne hinreichende Zulassung in Verkehr gebracht werden.

Mit einer Guidance für Mobile Medical Applications hat die FDA zwar einerseits etwas mehr Rechtssicherheit geschaffen, andererseits aber zugleich Fragen aufgeworfen, indem sie anhand von Beispielen erklärt, welche Apps zugelassen werden müssen und welche nicht.

Angesichts der Flut an Apps interessiert sich die FDA vor allem für Anwendungen mit Einbeziehung angeschlossener medizinischer Geräte oder zur Verarbeitung von mit medizinischen Geräten gewonnenen Daten, während viele Apps momentan nicht reguliert werden (Stichwort „Enforcement Discretion“), die streng genommen auch Medizinprodukte sind. Kann kein passendes Vergleichsbeispiel gefunden werden, wird die FDA für eine Einzelfallentscheidung kontaktiert. Für viele App-Hersteller, besonders im Bereich Fitness, Self-Monitoring, medizinische Informationsapps und Patient Records, entsteht damit in den USA ein rechtlicher Freiraum, jedoch ohne Garantie, dass die FDA ihre Politik zukünftig nicht ändert. In Europa gibt es bisher noch keine entsprechende Sonderregelung.

3.6 Fazit und Ausblick

Mobile Medical Apps werden sich auch in Zukunft weiter verbreiten. Der Smartphone-Markt ist nach wie vor in Bewegung und Marktanteile verschieben sich. Mit einer vollständigen Harmonisierung der verschiedenen App-Technologien ist auf absehbare Zeit nicht zu rechnen.

Mit den Cross-Compilation- und Cross-Platform-Ansätzen haben sich daher sehr vielversprechende Optionen für die App-Entwicklung ergeben. Auch wenn viele Tools inzwischen sehr ausgereift sind, gilt es doch, individuelle Vor- und Nachteile zu berücksichtigen. Die Wahl der Technologie sollte vom Anwendungsfall, also von der konkret geplanten App, abhängig gemacht werden. Oft sind die mit den Tools von Apple und Google entwickelten, klassischen nativen Apps nach wie vor eine gute Wahl.

Die Verantwortung für die Gesundheit von Patienten, Nutzern und Dritten spiegelt sich bei Mobile Medical Apps in einem Mehraufwand für Risikomanagement, Usability Engineering, Dokumentation, Verifizierung und Validierung. Insbesondere erhöht sich infolge wiederholter Verifizierungsdurchläufe auch der Testaufwand. Hier haben sich in den vergangenen Jahren Technologien etabliert, um Tests auf verschiedenen mobilen Zielplattformen automatisiert auszuführen, womit der manuelle Testaufwand drastisch reduziert werden kann.

Trotzdem kann je nach Zulassungsstrategie und Art der App der Dokumentationsmehraufwand für die Erfüllung der gesetzlichen Anforderungen den Aufwand für die eigentliche Entwicklung der App deutlich übertreffen.

Umso wichtiger ist es, die für die Bewältigung dieser Anforderungen notwendige Expertise in der Medizinproduktentwicklung mitzubringen und frühzeitig die richtigen Entscheidungen zu treffen.

4 Quellenverzeichnis

[1] FDA: http://www.fda.gov/

[2] FDA: Mobile Medical Applications; online abgerufen unter http://www.fda.gov/MedicalDevices/ProductsandMedicalProcedures/ConnectedHealth/MobileMedicalApplications/

[3] ZVEI-Leitfaden „Mobile Geräte und Apps in der Medizin“; online abgerufen unter http://www.zvei.org/Publikationen/Leitfaden-Mobile-Apps-in-der-Medizin.pdf

5 Glossar

CSS	Cascading Style Sheets
EMV	Elektromagnetische Verträglichkeit
FDA	Food and Drug Administration
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
JIT	Just in time
MDD	Medical Device Directive
SDK	Software Development Kit
ZVEI	Zentralverband Elektrotechnik- und Elektronikindustrie

Über die infoteam Software Gruppe

Die infoteam Software Gruppe realisiert seit fast 40 Jahren spezifische Softwarelösungen für ihre Kunden aus den Märkten Industry, Infrastructure, Life Science und Public Service. Das Kerngeschäft bilden die Teil- oder Gesamtentwicklung von Steuerungs- und Embedded-Software, Middleware und Anwendungssoftware – agil, modern und nach aktuellen Security-Anforderungen. Spezialdisziplinen sind u. a. normativ regulierte Software für den Einsatz in Medizin- und Laborgeräten (IVDR, MDR, FDA, ISO 13485, IEC 62304 etc.) sowie funktional sichere Software bis zur höchsten Sicherheitsstufe (IEC 61508, DIN EN 50128 etc.). Abgerundet wird das Leistungsportfolio durch langjährige Erfahrungen in den Bereichen Datenanalyse, KI und maschinelles Lernen.

Die infoteam Software Gruppe beschäftigt mehr als 300 Mitarbeiter und verfügt über Standorte und Tochtergesellschaften in Deutschland, Tschechien, der Schweiz und China. Stammsitz der Muttergesellschaft infoteam Software AG ist Bubenreuth bei Erlangen.

www.infoteam.de

Kontakt

infoteam Software AG
Am Bauhof 9 | 91088 Bubenreuth | Deutschland
Telefon: +49 9131 78 00-0
Telefax: +49 9131 78 00-50
info@infoteam.de | www.infoteam.de

Alle verwendeten Hard- und Softwarenamen sind Handelsmarken und/oder eingetragene Marken der jeweiligen Hersteller.

© 2014, infoteam Software AG.
Änderungen vorbehalten.