Drucksache.

Einen prägnanten Architekturüberblick anfertigen am Beispiel der Corona-Warn-App.

STEFAN ZÖRNER
embarc Architektur-Midsommar
Donnerstag, 24.06.2021 (Remote)

embarc
Software Consulting GmbH

1

Drucksache.

Einen prägnanten Architekturüberblick anfertigen am Beispiel der Corona-Warn-App.

Zusammenfassung:

Die Dokumentation Eurer Softwarearchitektur veraltet schnell? Deswegen fertigt Ihr gar keine an? Ich hätte da eine Lösung für Euch! In diesem Vortrag zeige ich am Beispiel der deutschen Corona-Warn-App wie Ihr die Architekturansätze eines Softwaresystems auf geringem Raum verdichtet und zugleich überzeugend darstellt. Statt sich in Word und Wiki zu verlieren erfasst Eure Zielgruppe einen solchen Überblick im Handumdrehen; bei Bedarf führt er Interessierte zielsicher in die nötigen Details.

Der Überblick ist mit geringem Aufwand anzufertigen und aktuell zu halten. Und sieht im Idealfall auch noch richtig gut aus! Neugierig geworden? Ein bisschen was zur Corona-Warn-App lernt Ihr obendrein.

2

Stefan Zörner

- Softwareentwickler + -architekt bei embarc in Hamburg
- Vorher oose, IBM, Mummert + Partner, Bayer AG, …

Schwerpunkte:

- Softwarearchitektur (Entwurf, Bewertung, Dokumentation)
- Java Technologien

Stefan.Zoerner@embarc.de

@StefanZoerner

xing.to/szr

embarc Software Consulting GmbH

3

Was erwartet Euch in diesem Vortrag?

- Welche **Inhalte** aus meiner Sicht in einen prägnanten **Architekturüberblick** gehören
- Ein **Überblick** über die Softwarearchitektur des deutschen **Corona-Warn-App-Systems** als durchgängiges Beispiel
- **Tipps und Tricks**, wie Ihr selber einen solchen Architekturüberblick anfertigt
- Ein paar Sachen zum **Mitmachen** und was schönes, lilafarbenes zum **Gucken**

4

Agenda

1 Wozu Architekturüberblicke?

2 Aufgabenbezogene Inhalte

3 Lösungsbezogene Inhalte

4 Tipps zum Anfertigen Eures Überblicks

5 Zusammenfassung + weitere Informationen

5

Agenda

1 **Wozu Architekturüberblicke?**

2 Aufgabenbezogene Inhalte

3 Lösungsbezogene Inhalte

4 Tipps zum Anfertigen Eures Überblicks

5 Zusammenfassung + weitere Informationen

1

6

Worum geht's?

Eurer Team tut sich schwer, die Eckpunkte seiner Software-Lösung zu kommunizieren.

- **Neue Teammitglieder**, die mitentwickeln wollen, finden sich in der Architektur nicht zurecht.

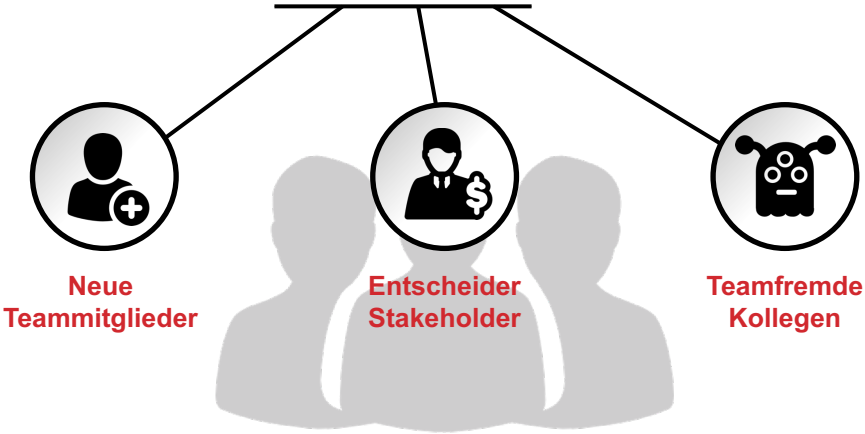
- **Entscheider** und andere Stakeholder haben Unsicherheiten oder geringes Vertrauen in Eure Lösung.

- **Teamfremde Kollegen** sind an Lösungsansätzen interessiert, finden aber keine oder sehr detaillierte Informationen.

7

Was ist ein Architekturüberblick?

Ein Architekturüberblick macht die zentralen Lösungsansätze Eurer Softwarearchitektur für Außenstehende nachvollziehbar.



8

Was ist Softwarearchitektur?

Softwarearchitektur :=

$\sum$ wichtige Entscheidungen

wichtig :=

- fundamental (betrifft viele)
- im weiteren Verlauf nur schwer zu ändern
- entscheidend für den Erfolg des Softwaresystems

9

Themen für Entscheidungen

Zerlegung

Welcher Architekturstil?
Wie zerfällt die Anwendung?
Teilsysteme, Module,
Komponentenbildung,
Abhängigkeiten ...

Zielumgebung

Wo läuft die Software?
Beim Endanwender, im
eigenen Rechenzentrum,
Cloud, Verteilung,
Virtualisierung ...

Technologie-Stack

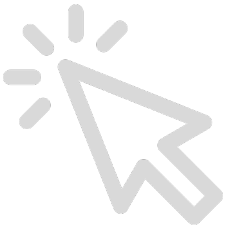
Was setzen wir ein?
Programmiersprache(n)
Bibliotheken, Frameworks,
Middleware,
Querschnittsthemen ....

Vorgehen

Wie arbeiten wir?
Planen, Entwickeln, Testen,
Bauen, Dokumentieren,
Ausliefern, Nachjustieren,
...

10

Kleine Interaktion in Mural …

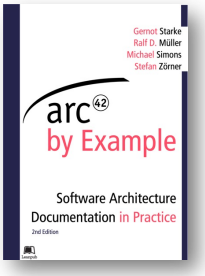


MURAL

11

Wäre arc42 was?

arc42 ist ein etablierter Gliederungsvorschlag für Architekturbeschreibungen.



1. Einführung und Ziele
1.1 Aufgabenstellung
1.2 Qualitätsziele
1.3 Stakeholder

2. Randbedingungen
2.1 Technische Randbedingungen
2.2 Organisatorische Randbedingungen
2.3 Konventionen

3. Kontextabgrenzung
3.1 Fachlicher Kontext
3.2 Technischer- oder Verteilungskontext

4. Lösungsstrategie

5. Bausteinsicht
5.1 Ebene 1
5.2 Ebene 2
...

6. Laufzeitsicht
6.1 Laufzeitszenario 1
6.2 Laufzeitszenario 2
...

7. Verteilungssicht
7.1 Infrastruktur Ebene 1
7.2 Infrastruktur Ebene 2
...

8. Konzepte
8.1 Fachliche Strukturen und Modelle
8.2 Typische Muster und Strukturen
8.3 Persistenz
8.4 Benutzungsoberfläche
...

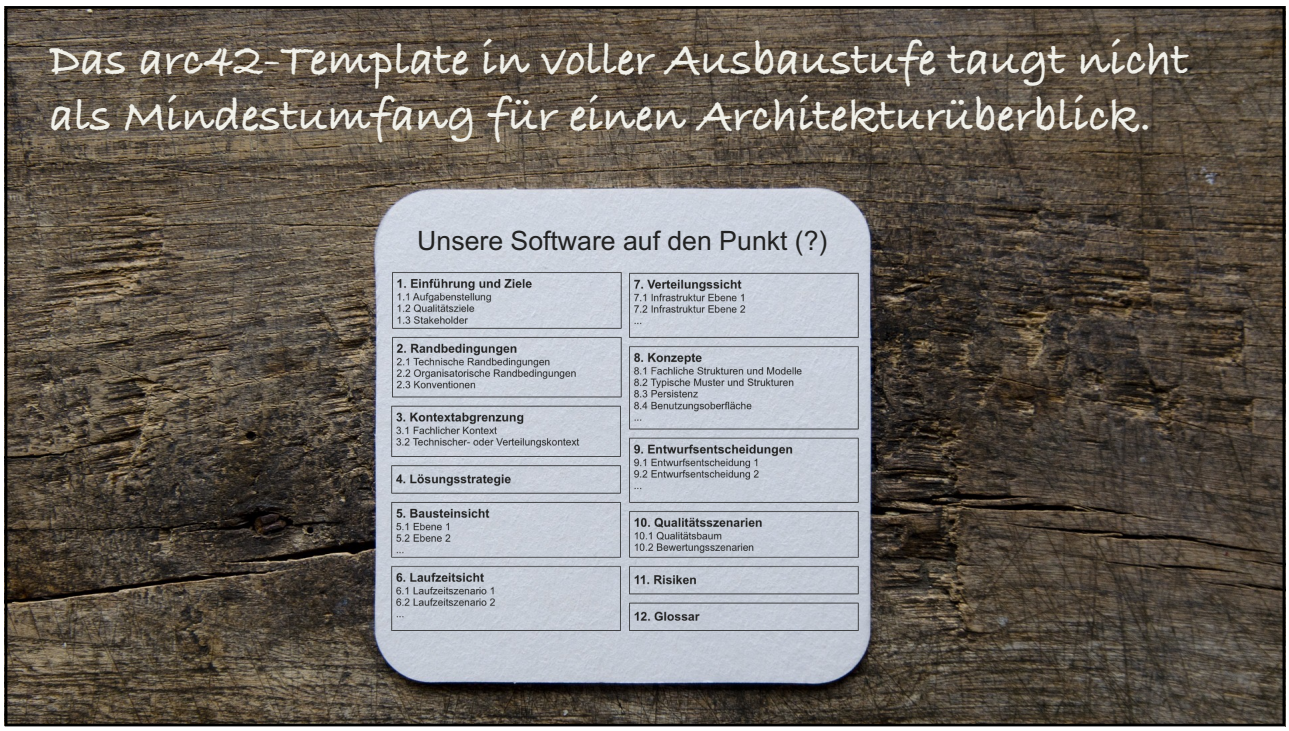
9. Entwurfsentscheidungen
9.1 Entwurfsentscheidung 1
9.2 Entwurfsentscheidung 2
...

10. Qualitätsszenarien
10.1 Qualitätsbaum
10.2 Bewertungsszenarien

11. Risiken

12. Glossar

12



13

Ein kompakterer Ansatz

- weitaus weniger Inhalte
- ausgehend von der Aufgabe
- nachvollziehbar durch Einbeziehung der Einflüsse

Aufgabe
- Mission Statement
- Kontextabgrenzung

Einflüsse
- Rahmenbedingungen
- Qualitätsziele

Lösungsansätze
- Architekturstil
- Technologie-Stack
- Konzepte
- Prinzipien
- Zerlegung
- ....

14

Agenda

1 Wozu Architekturüberblicke?

2 **Aufgabenbezogene Inhalte**

3 Lösungsbezogene Inhalte

4 Tipps zum Anfertigen Eures Überblicks

5 Zusammenfassung + weitere Informationen

2

15

Ein kompakter Ansatz

- wenige Inhalte
- **ausgehend von der Aufgabe**
- nachvollziehbar durch Einbeziehung der Einflüsse

Aufgabe

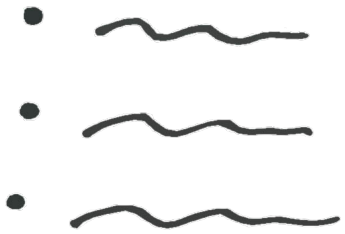
- Mission Statement
- Kontextabgrenzung

16

Mission Statement

Plakative Darstellung der Aufgabenstellung.

Form: Stichpunkte oder wenige (!) Sätze.



17

Mission Statement

Was ist die Corona-Warn-App?

Die Corona-Warn-App (kurz CWA) hilft Infektionsketten des SARS-CoV-2 (COVID-19-Auslöser) in Deutschland nachzuverfolgen und zu unterbrechen.

Die App basiert auf Technologien mit einem dezentralisierten Ansatz und informiert Personen, wenn sie mit einer infizierten Person in Kontakt standen.

Transparenz ist von entscheidender Bedeutung, um die Bevölkerung zu schützen und die Akzeptanz zu erhöhen.

nach www.coronawarn.app/de/

18

Euer Mission Statement

Leitfragen zur Erstellung

- Wozu ist das Software-System da?
- Was ist das zentrale Verkaufs- / Nutzungsargument? (“Claim”, “Slogan”)
- Wem nützt es?
- Was sind die wesentlichen Features des Systems?
- Wie unterscheidet es sich von Produkten der Mitbewerber, oder der Vorgängerversion?

19

Ein kompakter Ansatz

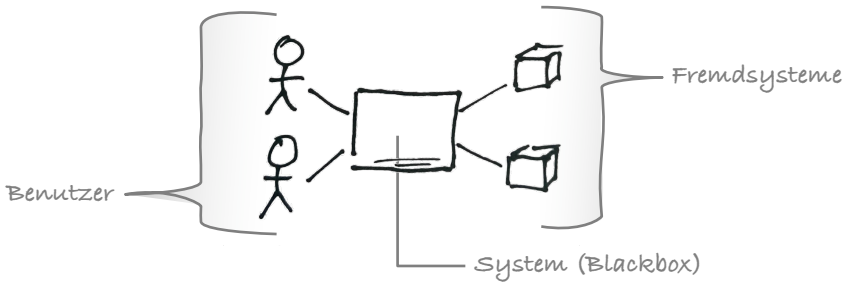
- wenige Inhalte
- **ausgehend von der Aufgabe**
- nachvollziehbar durch Einbeziehung der Einflüsse

20

Kontextabgrenzung

Abgrenzung des Systems und Visualisierung der Benutzer und Fremdsysteme, mit denen es interagiert.

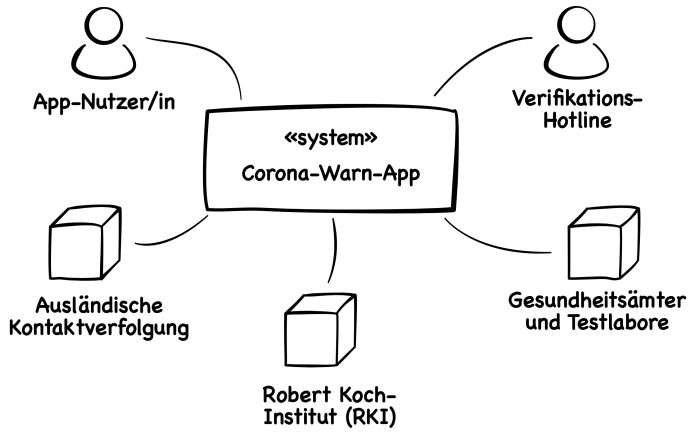
Form: Graphik, ergänzt um kurze Beschreibungen.



21

Fachliche Kontextabgrenzung

Das CWA-Gesamtsystem im Zusammenspiel mit den wichtigsten Benutzern und Fremdsystemen.



Legende:

Benutzer
CWA-System stellt Oberfläche bereit

Fremdsystem
CWA-System stellt Schnittstelle(n) bereit

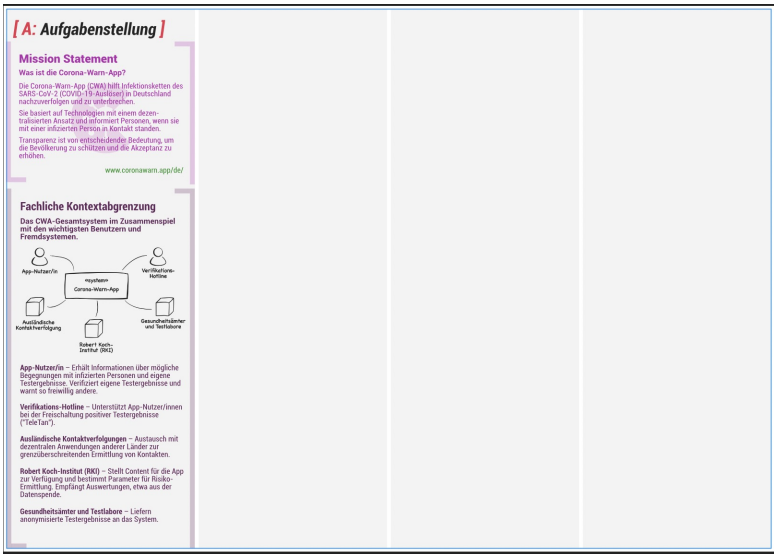
22

Fachliche Kontextabgrenzung, Akteure

Kurze Erläuterungen zu den Benutzern und Fremdsystemen:

Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Robert Koch-Institut (RKI)	Stellt Inhalte ("Content") für die App zur Verfügung und bestimmt Parameter für die Messung der Kontakte ("Risiko-Ermittlung"). Empfängt Auswertungen, etwa aus der Datenspende.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.

Überblicksflyer, 1. Spalte



[A: Aufgabenstellung]

Mission Statement

Was ist die Corona-Warn-App?

Die Corona-Warn-App (CWA) hilft Infektionsketten des SARS-CoV-2 (COVID-19-Auslöser) in Deutschland nachzuverfolgen und zu unterbrechen.

Sie basiert auf Technologien mit einem dezentralisierten Ansatz und informiert Personen, wenn sie mit einer infizierten Person in Kontakt standen.

Transparenz ist von entscheidender Bedeutung, um die Bevölkerung zu schützen und die Akzeptanz zu erhöhen.

www.coronawarn.app/de/

Fachliche Kontextabgrenzung

Das CWA-Gesamtsystem im Zusammenspiel mit den wichtigsten Benutzern und Fremdsystemen.

App-Nutzer/in – Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.

Verifikations-Hotline – Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("TeleTan").

Ausländische Kontaktverfolgungen – Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.

Robert Koch-Institut (RKI) – Stellt Content für die App zur Verfügung und bestimmt Parameter für Risiko-Ermittlung. Empfängt Auswertungen, etwa aus der Datenspende.

Gesundheitsämter und Testlabore – Liefern anonymisierte Testergebnisse an das System.

Ein kompakter Ansatz

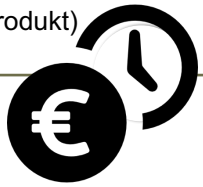
- wenige Inhalte
- ausgehend von der Aufgabe
- **nachvollziehbar durch Einbeziehung der Einflüsse**

Aufgabe
- Mission Statement
- Kontextabgrenzung

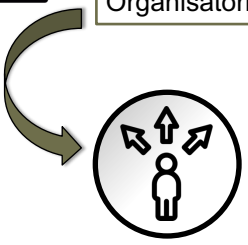
Einflüsse
- Rahmenbedingungen
- Qualitätsziele

Einflüsse auf Entscheidungen

Rahmenbedingungen
Technisch (z.B. Datenbankprodukt)
Organisatorisch (z.B. Team)



- schränken die Lösung ein
- schließen Optionen aus



Rahmenbedingungen CWA

Ausgewählte Maßgaben an Entwicklung und Betrieb sowie einige Informationen zum politischen Umfeld.

Technische Vorgaben

- Entwicklung von nativen, mobilen Clients für Android- und iOS-Smartphones
- Verfolgen eines dezentralen Ansatzes für die Datenspeicherung
- Einsatz des Exposure Notification Framework von Google und Apple
- Betrieb der Backend-Komponenten in der Open Telekom Cloud

Video zu dezentralem Ansatz

„Bundesregierung setzt bei Corona-App auf dezentrale Datenspeicherung"

Beitrag der Tagesschau vom 26.04.2020
https://www.tagesschau.de/multimedia/video/video-693083.html

Rahmenbedingungen CWA (Fortsetzung)

Organisatorischer Rahmen und Umfeld

- Auftraggeber: Deutsche Bundesregierung
- Entwicklung und Betrieb durch ein Konsortium aus zwei Auftragnehmern (SAP und Deutsche Telekom)
- Start der Entwicklung 04/2020
- enger Zeitrahmen (Apps zum Download verfügbar ab 06/2020)
- hoher politischer Druck, viele Parteien involviert (Ministerien, Behörden, RKI ...)
- große Medienaufmerksamkeit
- gewisse Skepsis innerhalb der Bevölkerung

Ein kompakter Ansatz

- wenige Inhalte
- ausgehend von der Aufgabe
- **nachvollziehbar durch Einbeziehung der Einflüsse**

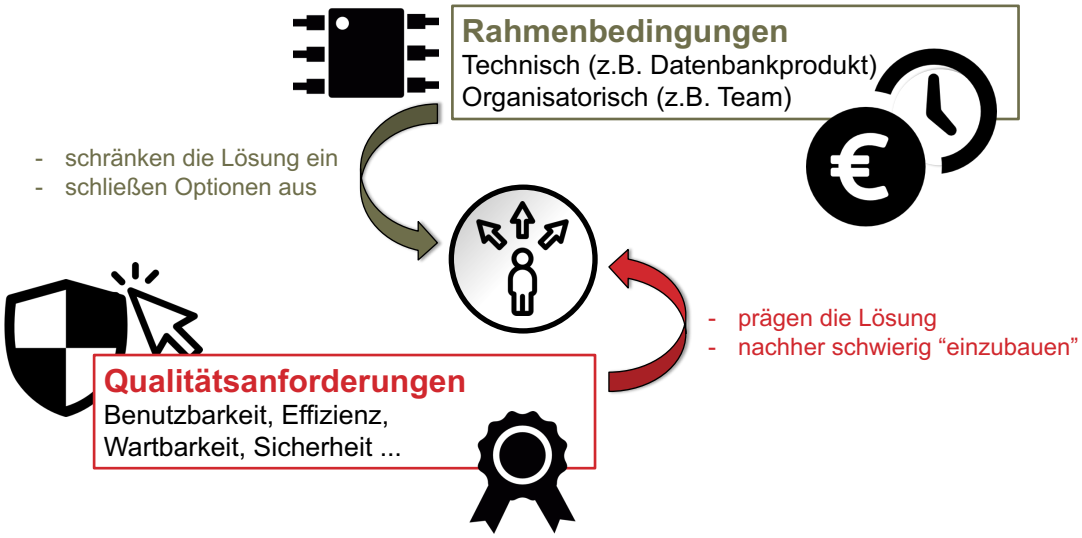
Aufgabe

- Mission Statement
- Kontextabgrenzung

Einflüsse

- Rahmenbedingungen
- Qualitätsziele

Einflüsse auf Entscheidungen

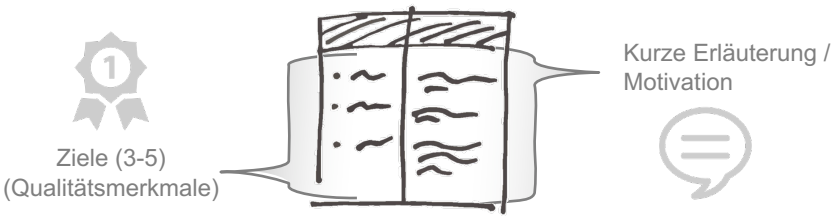


31

Architekturziele

Die wichtigsten (Top 3-5) an das System gestellten Qualitätsanforderungen („-ilities“), inkl. Motivation

Form: Stichpunkte oder (besser) Tabelle.



32

Qualitätsmerkmale

Begriffe nach ISO 25010

Benutzbarkeit (Usability)

Ist die Software intuitiv zu bedienen, leicht zu erlernen, attraktiv?

Portabilität (Portability)

Ist die Software leicht auf andere Zielumgebungen (z.B. anderes OS) übertragbar?

Funktionale Eignung (Functional Suitability)

Sind die berechneten Ergebnisse genau genug / exakt, ist die Funktionalität angemessen? ...

Effizienz (Performance)

Antwortet die Software schnell, hat sie einen hohen Durchsatz, einen geringen Ressourcenverbrauch? ...

Kompatibilität (Compatibility)

Ist die Software konform zu Standards, arbeitet sie gut mit anderen zusammen?

Zuverlässigkeit (Reliability)

Ist das System verfügbar, tolerant gegenüber Fehlern, nach Abstürzen schnell wieder hergestellt? ...

Sicherheit (Security)

Ist das System sicher vor Angriffen? Sind Daten und Funktion vor unberechtigtem Zugriff geschützt? ...

Wartbarkeit (Maintainability)

Ist die Software leicht zu ändern, erweitern, testen, verstehen? Lassen sich Teile wiederverwenden? ...

33

Kleine Interaktion in Mural …

MURAL

34

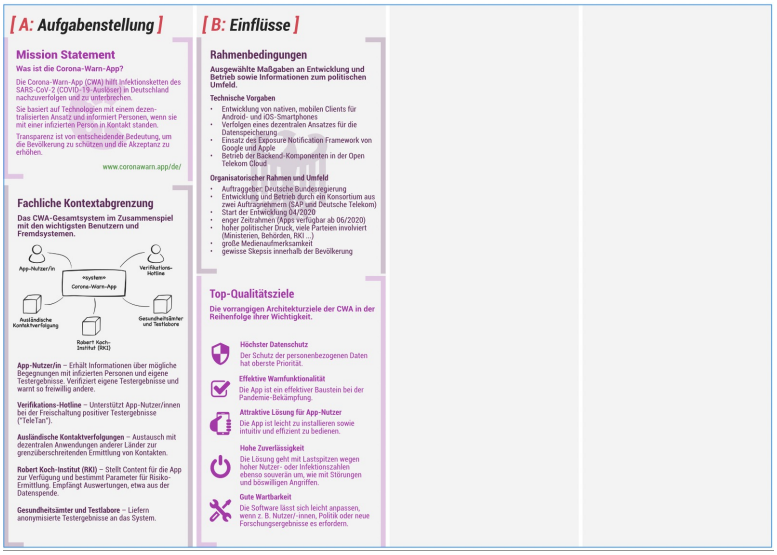
Top-Qualitätsziele

Die vorrangigen Architekturziele der CWA in der Reihenfolge ihrer Wichtigkeit.

	Ziel	Beschreibung
	Höchster Datenschutz	Der Schutz der personenbezogenen Daten hat oberste Priorität. *(Sicherheit)*
	Effektive Warnfunktionalität	Die App ist ein effektiver Baustein bei der Pandemie-Bekämpfung. *(Funktionale Eignung)*
	Attraktive Lösung für App-Nutzer	Die App ist leicht zu installieren sowie intuitiv und effizient zu bedienen. *(Benutzbarkeit)*
	Hohe Zuverlässigkeit	Die Lösung geht mit Lastspitzen wegen hoher Nutzer- oder Infektionszahlen ebenso souverän um, wie mit Störungen und böswilligen Angriffen. *(Zuverlässigkeit)*
	Gute Wartbarkeit	Die Software lässt sich leicht anpassen, wenn z. B. Nutzer/-innen, Politik oder neue Forschungsergebnisse es erfordern. *(Wartbarkeit/Erweiterbarkeit)*

35

Überblicksflyer, 1. + 2. Spalte



36

Agenda

1 Wozu Architekturüberblicke?

2 Aufgabenbezogene Inhalte

3 **Lösungsbezogene Inhalte**

4 Tipps zum Anfertigen Eures Überblicks

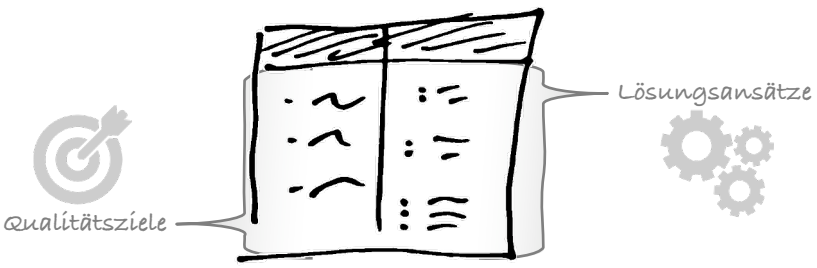
5 Zusammenfassung + weitere Informationen

3

Lösungsstrategie (Tabelle)

Qualitätsziele und zugeordnete high-level Lösungsansätze.

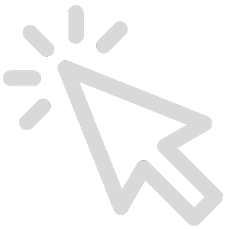
Form: Tabelle/Zuordnung ([Ziele | Lösungsansätze]).



Typische Lösungsansätze

39

Kleine Interaktion in Mural …



MURAL

40

Entscheidende Lösungsansätze

Die aufgelisteten Prinzipien, Muster, Konzepte und Technologien adressieren die jeweiligen Top-Qualitätsziele und begünstigen deren Erreichung.

Qualitätsziel: Höchster Datenschutz

- **Datensparsamkeit** (nur minimale, notwendige Informationen)
- **Dezentraler Ansatz**, Speicherung der Daten lokal beim App-Nutzer
- Verschlüsselung aller Bewegungsdaten
- Teilen von Daten nur nach Aufforderung
- Transparente Entwicklung (**Quelltexte** und Konzepte **einsehbar**)

Entscheidende Lösungsansätze (2/3)

Qualitätsziel: Effektive Warnfunktionalität

- **Digitale Abläufe** zur Risikowarnung
- Unterstützung der **gängigsten Smartphones** (mehr als 90% aller Geräte)
- Verwendung des Exposure Notification Frameworks

Qualitätsziel: Attraktive Lösung für App-Nutzer

- **Native Apps für iOS** und Android
- Verteilung durch die jeweiligen Stores
- **Übersichtliche Gestaltung**, simple Bedienung
- **Geringer Ressourcen-Verbrauch**, u.a. durch Bluetooth Low Energy und Datensparsamkeit

Entscheidende Lösungsansätze (3/3)

Qualitätsziel: Hohe Zuverlässigkeit

- **Robuste Lösung**, Apps erfassen Begegnungen ohne Verbindung zum Backend
- Backend in der **Public Cloud**, Orchestrierung mit Kubernetes
- **Modularisierung** des Backends in unabhängige, skalierbare Services
- Bereitstellung von zu lesenden Daten über **Content Delivery Network**

Qualitätsziel: Gute Wartbarkeit

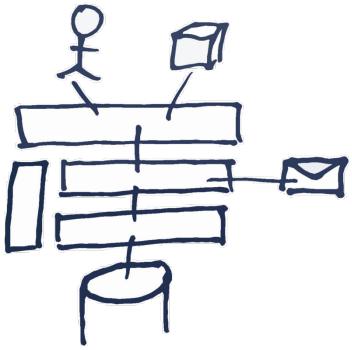
- Fachlich getriebene Zerlegung der Apps
- **Modularisierung** des Backends in Microservices mit separater Datenhaltung
- Verwendung verbreiteter Standard- und Open Source-Bibliotheken
- Entwicklung als Open Source, **gute Dokumentation**
- Tool-gestützte Überwachung der Code-Qualität

43

Informelles Überblicksbild

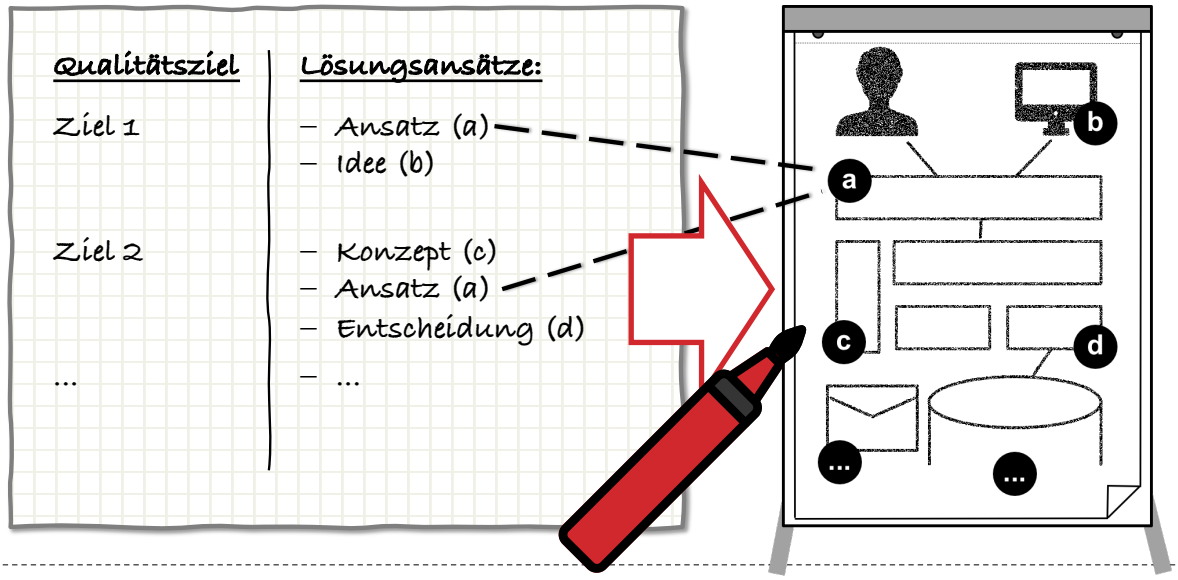
Visualisierung der Lösung mit Betonung der zentralen Architekturansätze (z.B. Stil, Muster, Struktur ...)

Form: Abbildung (eher kein UML ;-)



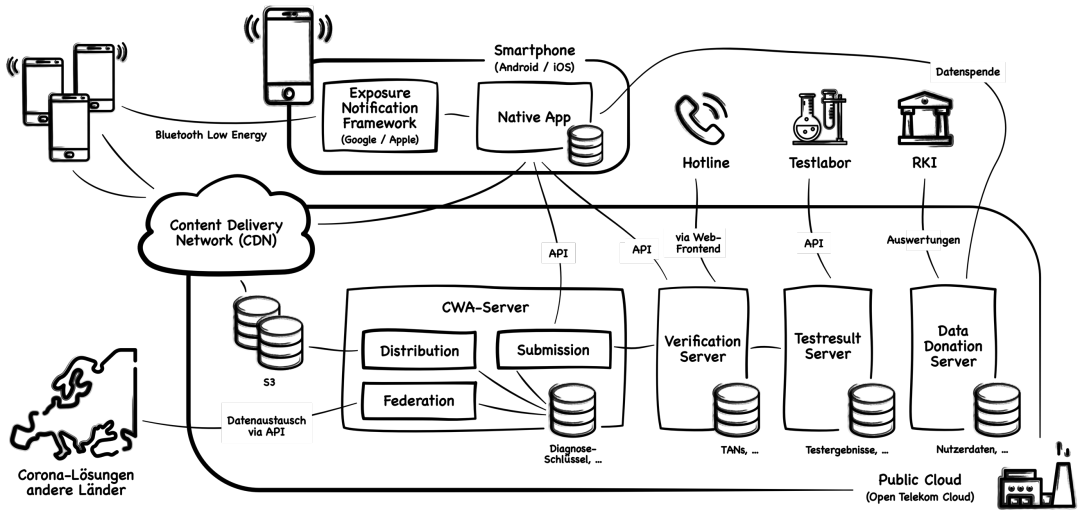
44

Überblicksbild aus der Lösungsstrategie ableiten



45

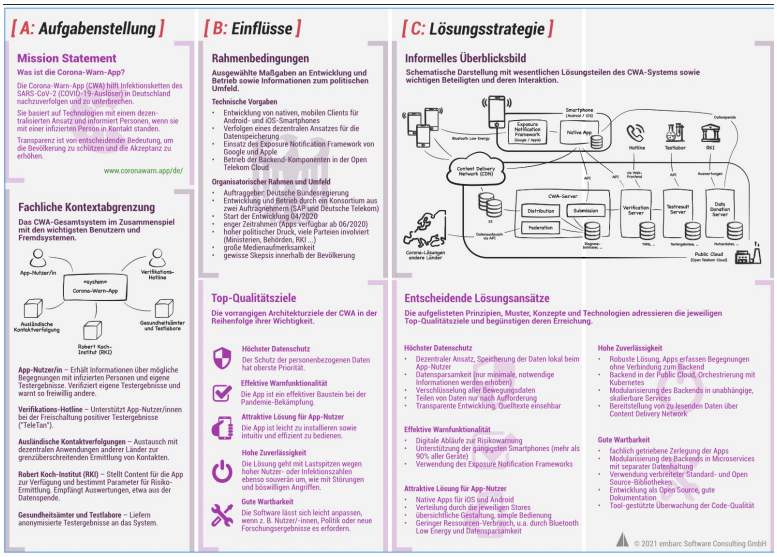
Informelles Überblicksbild



Quelle der Abbildung: S. Zörner, F. Sippach: „So gehen Architektur-Reviews! Entlang der Corona-Warn-App", OOP 2021

46

Überblicksflyer, 1. Seite (innen) komplett



[A: Aufgabenstellung]

Mission Statement

Was ist die Corona-Warn-App?

Die Corona-Warn-App (CWA) hilft Infektionsketten des SARS-CoV-2 (COVID-19-Auslöser) in Deutschland nachzuverfolgen und zu unterbrechen.

Sie basiert auf Technologien mit einem dezentralisierten Ansatz und informiert Personen, wenn sie mit einer infizierten Person in Kontakt standen.

Transparenz ist von entscheidender Bedeutung, um die Bevölkerung zu schützen und die Akzeptanz zu erhöhen.

www.coronawarn.app/de/

Fachliche Kontextabgrenzung

Das CWA-Gesamtsystem im Zusammenspiel mit den wichtigsten Benutzern und Fremdsystemen.

App-Nutzer/in – Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.

Verifikations-Hotline – Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("TeleTan").

Ausländische Kontaktverfolgungen – Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.

Robert Koch-Institut (RKI) – Stellt Content für die App zur Verfügung und bestimmt Parameter für Risiko-Ermittlung. Empfängt Auswertungen, etwa aus der Datenspende.

Gesundheitsämter und Testlabore – Liefern anonymisierte Testergebnisse an das System.

[B: Einflüsse]

Rahmenbedingungen

Ausgewählte Maßgaben an Entwicklung und Betrieb sowie Informationen zum politischen Umfeld.

Technische Vorgaben

- Entwicklung von nativen, mobilen Clients für Android- und iOS-Smartphones
- Verfolgen eines dezentralen Ansatzes für die Datenspeicherung
- Einsatz des Exposure Notification Framework von Google und Apple
- Betrieb der Backend-Komponenten in der Open Telekom Cloud

Organisatorischer Rahmen und Umfeld

- Auftraggeber: Deutsche Bundesregierung
- Entwicklung und Betrieb durch ein Konsortium aus zwei Auftragnehmern (SAP und Deutsche Telekom)
- Start der Entwicklung 04/2020
- enger Zeitrahmen (Apps verfügbar ab 06/2020)
- hoher politischer Druck, viele Parteien involviert (Ministerien, Behörden, RKI ...)
- große Medienaufmerksamkeit
- gewisse Skepsis innerhalb der Bevölkerung

Top-Qualitätsziele

Die vorrangigen Architekturziele der CWA in der Reihenfolge ihrer Wichtigkeit.

Höchster Datenschutz
Der Schutz der personenbezogenen Daten hat oberste Priorität.

Effektive Warnfunktionalität
Die App ist ein effektiver Baustein bei der Pandemie-Bekämpfung.

Attraktive Lösung für App-Nutzer
Die App ist leicht zu installieren sowie intuitiv und effizient zu bedienen.

Hohe Zuverlässigkeit
Die Lösung geht mit Lastspitzen wegen hoher Nutzer- oder Infektionszahlen ebenso souverän um, wie mit Störungen und böswilligen Angriffen.

Gute Wartbarkeit
Die Software lässt sich leicht anpassen, wenn z. B. Nutzer/-innen, Politik oder neue Forschungsergebnisse es erfordern.

[C: Lösungsstrategie]

Informelles Überblicksbild

Schematische Darstellung mit wesentlichen Lösungsteilen des CWA-Systems sowie wichtigen Beteiligten und deren Interaktion.

Entscheidende Lösungsansätze

Die aufgelisteten Prinzipien, Muster, Konzepte und Technologien adressieren die jeweiligen Top-Qualitätsziele und begünstigen deren Erreichung.

Höchster Datenschutz
- Dezentraler Ansatz, Speicherung der Daten lokal beim App-Nutzer
- Datensparsamkeit (nur minimale, notwendige Informationen werden erhoben)
- Verschlüsselung aller Bewegungsdaten
- Teilen von Daten nur nach Aufforderung
- Transparente Entwicklung, Quellcode einsehbar

Effektive Warnfunktionalität
- Digitale Abläufe zur Risikoanalyse
- Unterstützung der gängigsten Smartphones (mehr als 90% aller Geräte)
- Verwendung des Exposure Notification Frameworks

Attraktive Lösung für App-Nutzer
- Native Apps für iOS und Android
- Verteilung durch die jeweiligen Stores
- übersichtliche Gestaltung, simple Bedienung
- Geringer Ressourcen-Verbrauch, u.a. durch Bluetooth Low Energy und Stromsparsamkeit

Hohe Zuverlässigkeit
- Robuste Lösung, Apps erfassen Begegnungen ohne Verbindung zum Backend
- Backend in der Public Cloud, Orchestrierung mit Kubernetes
- Modularisierung des Backends in unabhängige, skalierbare Services
- Bereitstellung von zu lesenden Daten über Content Delivery Network

Gute Wartbarkeit
- fachlich getriebene Zerlegung der Apps
- Modularisierung des Backends in Microservices mit separater Datenhaltung
- Verwendung verbreiteter Standard- und Open Source-Bibliotheken
- Entwicklung als Open Source, gute Dokumentation
- Tool-gestützte Überwachung der Code-Qualität

© 2021 embarc Software Consulting GmbH

47

Ein kompakter Ansatz

- wenige Inhalte
- ausgehend von der Aufgabe
- nachvollziehbar durch Einbeziehung der Einflüsse

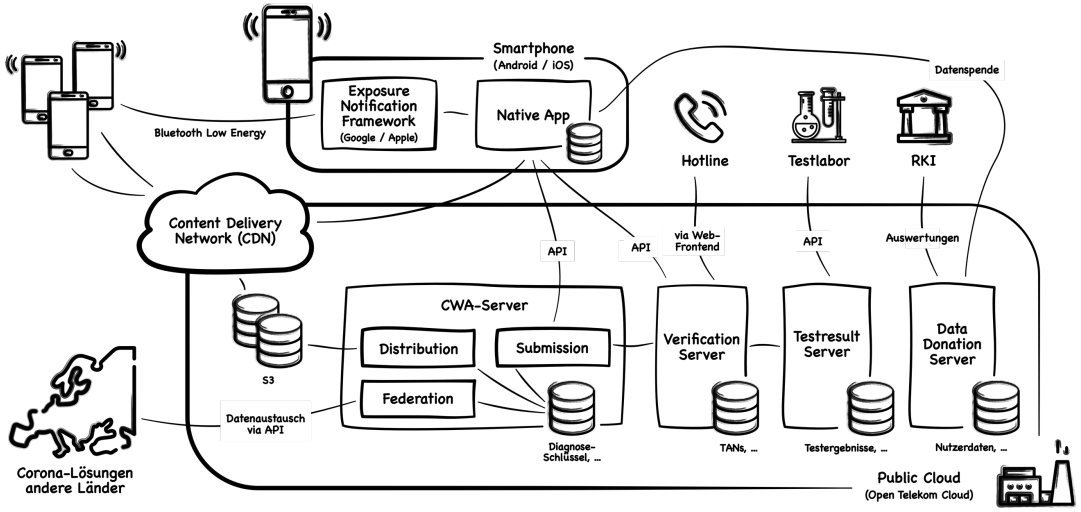
Aufgabe
- Mission Statement
- Kontextabgrenzung

Einflüsse
- Rahmenbedingungen
- Qualitätsziele

Lösungsansätze
- Architekturstil
- Technologie-Stack
- Konzepte
- Prinzipien
- Zerlegung
- ....

48

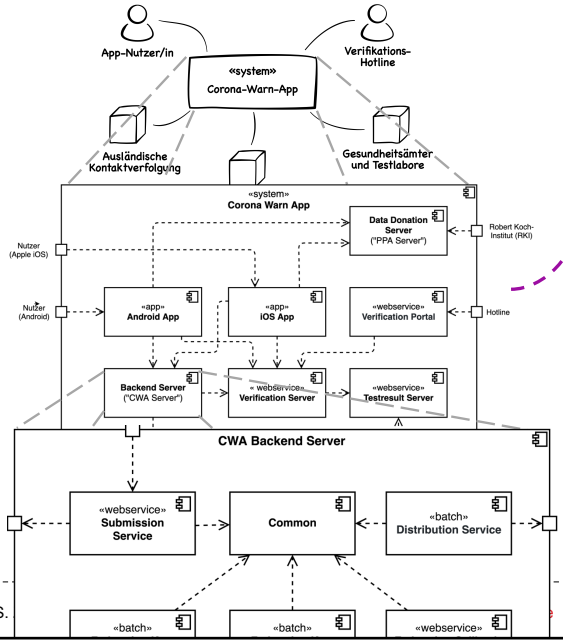
Informelles Überblicksbild



Quelle der Abbildung: S. Zörner, F. Sippach: „So gehen Architektur-Reviews! Entlang der Corona-Warn-App", OOP 2021

49

Tatsächliche Zerlegung im Quelltext



„Bausteinsicht, Ebene 1"

GitHub-Repositories

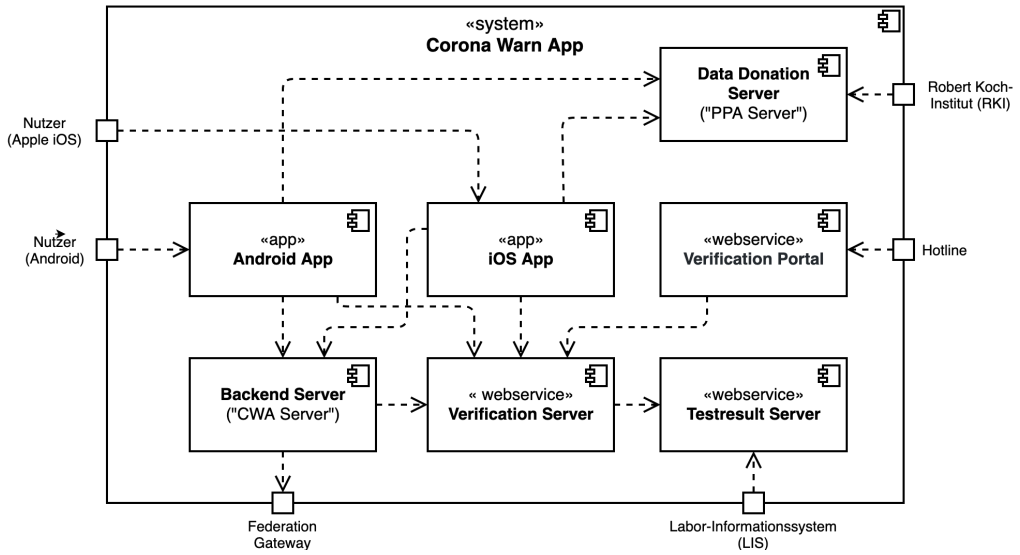
https://github.com/corona-warn-app/...

- cwa-app-android
- cwa-app-ios
- cwa-ppa-server („Data Donation")
- cwa-server („Backend")
- cwa-testresult-server
- cwa-verification-server
- cwa-verification-portal

CWA-Server (Backend)
Bausteinsicht, Ebene 2

50

Grobe Zerlegung (Bausteinsicht, Ebene 1)



51

Grobe Zerlegung, Bausteine auf Ebene 1

Baustein	Wesentliche Aufgabe
iOS App	Native App für iPhones. Nutzt das GAEN-System von Google und Apple zur Kontaktermittlung. Ermittelt das Infektionsrisiko anhand von Diagnoseschlüsseln.
Android App	Native Android App, analog zur iOS App.
Backend Server (aka „CWA Server")	Nimmt die Diagnoseschlüssel positiv getesteter Nutzer entgegen und teilt sie mit anderen Nutzern über ein CDN. Interagiert mit den Kontaktverfolgungen anderer Länder („Federation").
Testresult Server	Empfängt und speichert die Testergebnisse von angeschlossenen Laboren.
Verification Server	Sichert ab, dass ein Nutzer zugestimmt hat, seinen positiven Test zu melden, und dass das Labor tatsächlich positiv getestet hat.
Verification Portal	Ermöglicht die Erzeugung von teleTANs im Verification Server über ein einfaches Browser-Frontend.
Data Donation Server (aka „PPA Server")	Nimmt Nutzerdaten bei aktivierter Datenspende entgegen und speichert sie, ohne Rückschlüsse auf individuelle Personen zuzulassen. (PPA = Privacy Preserving Analytics)

52

Ein kompakter Ansatz

- wenige Inhalte
- ausgehend von der Aufgabe
- nachvollziehbar durch Einbeziehung der Einflüsse

Aufgabe
- Mission Statement
- Kontextabgrenzung

Einflüsse
- Rahmenbedingungen
- Qualitätsziele

Lösungsansätze
- Architekturstil
- Technologie-Stack
- Konzepte
- Prinzipien
- Zerlegung
- ....

Ausschnitt Technologie-Stack (1/3)

Die CWA-Lösung verwendet diverse Programmiersprachen, Bibliotheken, Frameworks und Middleware.

Client (Native Apps)

- iOS App programmiert in **Swift**
- Android App programmiert in **Kotlin**
- Persistenz mit **SQLite**
- Verteilung über **Google Play** (Android) und **App Store** (Apple iOS)

Swift Kotlin SQLite GET IT ON Google Play Download on the App Store

Ausschnitt Technologie-Stack (2/3)

Server (Backend)

- Services programmiert in **Java** mit **Spring** Boot, Spring Cloud, Spring Data ...
- weitere Open Source Bibliotheken (u.a. Lombok, Guava, Liquibase, **Micrometer** ...)
- Kommunikation mit REST/**OpenAPI** und protobuf
- Persistenz mit **PostgreSQL**

Java spring boot Micrometer OpenAPI PostgreSQL

55

Ausschnitt Technologie-Stack (3/3)

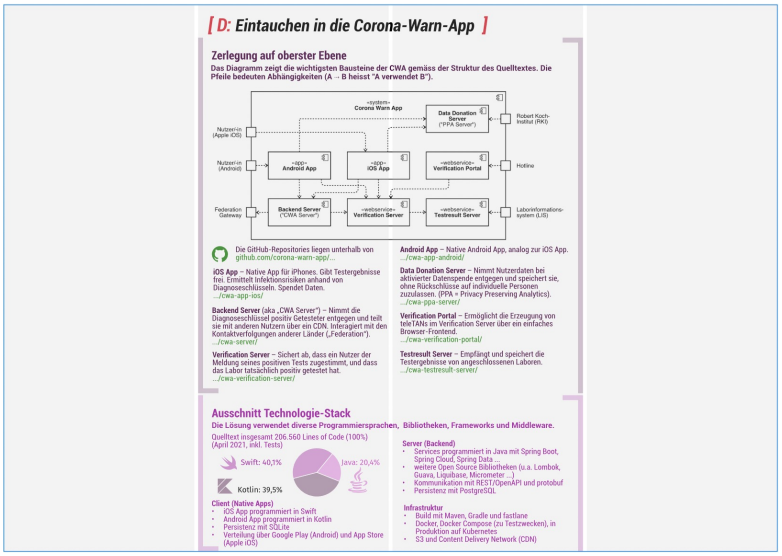
Infrastruktur

- Build mit **Maven**, **Gradle** und fastlane
- **Docker**, Docker Compose (zu Testzwecken), in Produktion auf **Kubernetes**
- S3 und Content Delivery Network (CDN)
- Betrieb in Public Cloud (**Open Telekom Cloud**)

maven Gradle docker kubernetes Open Telekom Cloud

56

Überblicksflyer, 2. Seite, Mitte



57

Agenda

1 Wozu Architekturüberblicke?
2 Aufgabenbezogene Inhalte
3 Lösungsbezogene Inhalte
4 **Tipps zum Anfertigen Eures Überblicks**
5 Zusammenfassung + weitere Informationen

4

58

Wann anfertigen? (1/2)

Einen Überblick in Form der gezeigten Inhalte für ein **bestehendes Softwaresystem** anzufertigen ist problemlos möglich.

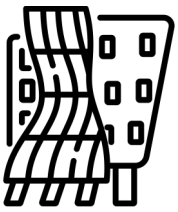
3 Tipps zur Erstellung („vorhandenes System“):

- startet mit der Kontextabgrenzung – hilft Euch den Scope festzulegen
- Zentrales Element: Lösungsstrategie (Qualitätsziele + Zuordnung der Lösungsansätze) – das gemeinsam im Team in einem Workshop erstellen
- Beschränkung des Gesamtumfangs, um den Überblick prägnant zu halten (siehe z.B. Flyer)

59

Wann anfertigen? (2/2)

Die gezeigten Inhalte helfen auch super beim Start von **neuen Vorhaben**. Das Ergebnis des Überblicks kann dann die **Architekturvision** Eures Teams sein.



3 Tipps zur Erstellung („neues System“):

- Startet mit den Rahmenbedingungen und Qualitätszielen – Kontextabgrenzung im Anschluss.
- Fertigt die Teile einzeln und iterativ an – insbesondere die Lösungsstrategie „wächst“.
- Haltet insbesondere Architekturentscheidungen einzeln fest (Stichwort: Architecture Decision Records).

60

Geeignete Form wählen (Stärken und Schwächen)

	Dokument	Foliensatz	Wiki	Architekturwand	Poster/Flyer	Video
Initialer Aufwand	niedrig, bei Start mit wenigen Zutaten und guter Struktur	niedrig, bei Start mit wenigen Folien und guter Struktur	mittel, ggf. Auswahl/ Einrichtung eines Produktes nötig	mittel, geeignete freie Wand und passende Kultur nötig	Anspruch an erste Auflage oft bereits sehr hoch	mittel, vor allem die Technik zu beherrschen kostet Zeit
Ändern und erweitern im nachhinein	leicht zu ändern und um Zutaten zu ergänzen, bei passendem Tooling	leicht zu ändern und zu ergänzen, Versionierung ggf. schwierig	leicht, Versionen und Verfolgen von Änderungen aber je nach Produkt	jederzeit möglich, aber Änderungen nachhalten schwierig	schwierig, da Platz begrenzt und Produktion aufwändig	im Nachhinein zu ändern oder um neue Inhalte zu erweitern ist aufwändig
Akzeptanz bei Entwicklern	gering, OK zum Lesen, falls prägnant und zielgruppen-gerecht	allein nur mittel, in Präsentationen / bei Durchsprachen höher	eher hoch, gleich-zeitig Vorurteile wegen häufiger „Verrottung“	hoch, da die Wand zu Feedback und Mitarbeit einlädt	hoch, falls gut gemacht. Spannendes, ungewohntes Format	hoch, wird gerne angesehen und geteilt, vor allem wenn lebendig gemacht
Akzeptanz bei Managern	hoch, entspricht oft der Erwartung	allein nur mittel, um Präsenta-tionen zu unter-stützen höher	gering, etwas besser bei ziel-gruppengerechten Einstiegsseiten	mittel, wird eher als Arbeitsmittel der Entwicklung gesehen	hoch, falls grafisch ansprechend und klar gestaltet	mittel, sollte auf jeden Fall kurz und zielgruppen-gerecht sein
Kommunikation in räumlich verteilten Teams	Verteilung zwar einfach, aber fördert für sich allein nicht den Austausch	mittel. Folien allein oft nicht aussage-kräftig, zusätzliche Präsentationen aufwändig	für Kollaboration in verteilten Teams vergleichsweise gut geeignet	schlecht, ggf. ist eine Verbreitung über Fotos möglich	sind gut an Loka-tionen zu verteilen, Flyer ebenso, Feedback schwierig	Gut zu teilen, auch hier aber Rückmeldungen schwierig

Legende Einschätzung:

- positiv
- neutral
- negativ

61

Gegen das Veralten

Veraltete Inhalte sind eines der häufigsten Probleme in Architekturdokumentation.

Das könnt Ihr bei Eurem Überblick dagegen tun:

- Inhalte zeigen, die relativ **stabil** sind
- bei einzelnen Inhalten jeweils nur das Wichtigste zeigen, und darauf hinweisen („ausgewählte …“, „Ausschnitt …“)
- angeben, welchen **Stand** der Überblick zeigt (z.B. Stand: April 2021)
- nur den **Abschluss** des Überblicks nutzen, um einen **Ausblick** auf weitere Entwicklungen zu geben.

Der gezeigte CWA-Überblick nutzt alle diese Möglichkeiten.

62

Einen runden Abschluss finden

Idealerweise endet Eurer Architekturüberblick nicht einfach irgendwo, nachdem es immer detaillierter wurde, und lässt den Leser allein.

Nutzt den Abschluss des Überblicks, um …

- die Lösung zu **reflektieren** (vor allem: Schwächen, Kompromisse)
- Hinweise zu **geplanten Entwicklungen** zu geben
- auf **detailliertere** Informationen zu verweisen (API, Quelltext, Demo-System …)
- zu **Rückmeldungen** einzuladen

63

Ausgewählte Kompromisse (1/2)

Der Entwurf der CWA geht bewusst Trade-offs ein und balanciert Qualitätsziele aus.

Explizite Freigabe positiver Testergebnisse durch Nutzer/in erforderlich

+ erhöht Vertrauen in die Lösung
- reduziert effektive Warnfunktionalität

Verteilte Anwendung auf dem Backend

+ gut für Datenschutz (Trennung der Daten)
+ verfügbar(er) im Falle von Teilausfällen
- schwieriger zu entwickeln und zu betreiben

64

Ausgewählte Kompromisse (2/2)

Ausliefern der Diagnoseschlüssel über CDN im Batch, Aktualisierung durch Apps in Intervallen

+ spart Ressourcen, vor allen an den Endgeräten
+ robust, erhöht Zuverlässigkeit
– Zeitverzögerung bei Risikoermittlung

Vergleichsweise hohe Kopplung der Microservices (tw. synchrone Kommunikation, tw. geteilte Daten)

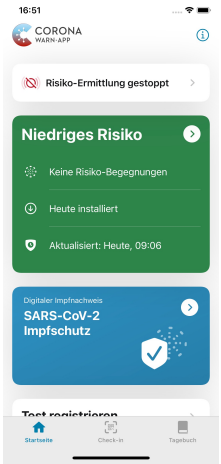
+ einfacher umzusetzen, schneller am Markt
– erschwert unabhängige Entwicklung
– reduziert oder behindert Zuverlässigkeit

65

Nächste Schritte für die CWA

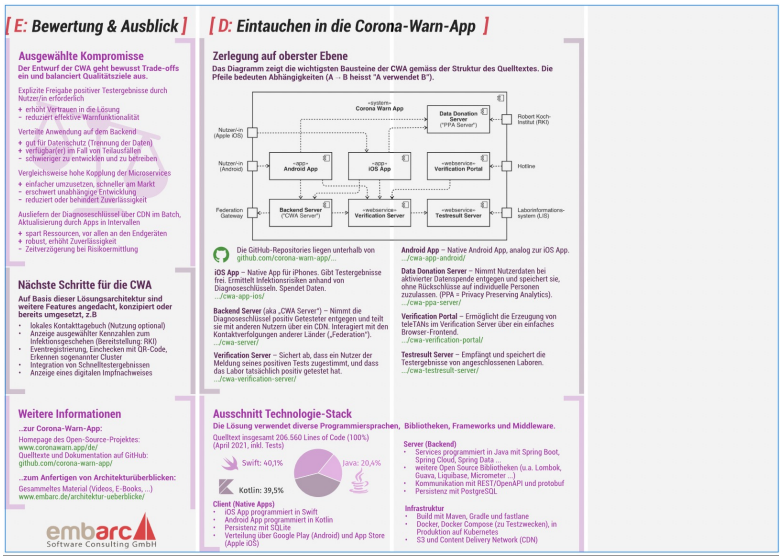
Auf Basis der Lösungsarchitektur wurden weitere Features angedacht, konzipiert oder auch bereits umgesetzt, z.B.

- lokales **Kontakttagebuch** (Nutzung optional)
- Anzeige ausgewählter **Kennzahlen** zum Infektionsgeschehen (Bereitstellung: RKI)
- **Eventregistrierung**, Einchecken mit QR-Code, Erkennen sogenannter Cluster
- Integration von **Schnelltestergebnissen**
- Anzeige eines **digitalen Impfnachweises**



66

Überblicksflyer, 2. Seite, Ausblick



[E: Bewertung & Ausblick]

Ausgewählte Kompromisse

Der Entwurf der CWA geht bewusst Trade-offs ein und balanciert Qualitätsziele aus.

Explizite Freigabe positiver Testergebnisse durch Nutzer/in erforderlich
+ erhöht Vertrauen in die Lösung
- reduziert effektive Warnfunktionalität

Verteilte Anwendung auf dem Backend
+ gut für Datenschutz (Trennung der Daten)
+ verfügbar(er) im Fall von Teilausfällen
- schwieriger zu entwickeln und zu betreiben

Vergleichsweise hohe Kopplung der Microservices
+ einfacher umzusetzen, schneller am Markt
- erschwert unabhängige Entwicklung
- reduziert oder behindert Zuverlässigkeit

Ausliefern der Diagnoseschlüssel über CDN im Batch, Aktualisierung durch Apps in Intervallen
+ spart Ressourcen, vor allen an den Endgeräten
+ robust, erhöht Zuverlässigkeit
- Zeitverzögerung bei Risikoermittlung

Nächste Schritte für die CWA

Auf Basis dieser Lösungsarchitektur sind weitere Features angedacht, konzipiert oder bereits umgesetzt, z.B

- lokales Kontakttagebuch (Nutzung optional)
- Anzeige ausgewählter Kennzahlen zum Infektionsgeschehen (Bereitstellung: RKI)
- Eventregistrierung, Einchecken mit QR-Code, Erkennen sogenannter Cluster
- Integration von Schnelltestergebnissen
- Anzeige eines digitalen Impfnachweises

Weitere Informationen

...zur Corona-Warn-App:

Homepage des Open-Source-Projektes:
www.coronawarn.app/de/
Quelltexte und Dokumentation auf GitHub:
github.com/corona-warn-app/

...zum Anfertigen von Architekturüberblicken:

Gesammeltes Material (Videos, E-Books, ...)
www.embarc.de/architektur-ueberblicke/

embarc Software Consulting GmbH

[D: Eintauchen in die Corona-Warn-App]

Zerlegung auf oberster Ebene

Das Diagramm zeigt die wichtigsten Bausteine der CWA gemäss der Struktur des Quelltextes. Die Pfeile bedeuten Abhängigkeiten (A → B heisst "A verwendet B").

Die GitHub-Repositories liegen unterhalb von github.com/corona-warn-app/...

iOS App – Native App für iPhones. Gibt Testergebnisse frei. Ermittelt Infektionsrisiken anhand von Diagnoseschlüsseln. Spendet Daten.
.../cwa-app-ios/

Backend Server (aka „CWA Server") – Nimmt die Diagnoseschlüssel positiv Getesteter entgegen und teilt sie mit anderen Nutzern über ein CDN. Interagiert mit den Kontaktverfolgungen anderer Länder („Federation").
.../cwa-server/

Verification Server – Sichert ab, dass ein Nutzer der Meldung seines positiven Tests zugestimmt, und dass das Labor tatsächlich positiv getestet hat.
.../cwa-verification-server/

Android App – Native Android App, analog zur iOS App.
.../cwa-app-android/

Data Donation Server – Nimmt Nutzerdaten bei aktiverter Datenspende entgegen und speichert sie, ohne Rückschlüsse auf individuelle Personen zuzulassen. (PPA = Privacy Preserving Analytics).
.../cwa-ppa-server/

Verification Portal – Ermöglicht die Erzeugung von teleTANs im Verification Server über ein einfaches Browser-Frontend.
.../cwa-verification-portal/

Testresult Server – Empfängt und speichert die Testergebnisse von angeschlossenen Laborem.
.../cwa-testresult-server/

Ausschnitt Technologie-Stack

Die Lösung verwendet diverse Programmiersprachen, Bibliotheken, Frameworks und Middleware.

Quelltext insgesamt 206.560 Lines of Code (100%)
(April 2021, inkl. Tests)

Client (Native Apps)
- iOS App programmiert in Swift
- Android App programmiert in Kotlin
- Persistenz mit SQLite
- Verteilung über Google Play (Android) und App Store (Apple iOS)

Server (Backend)
- Services programmiert in Java mit Spring Boot, Spring Cloud, Spring Data ...
- weitere Open Source Bibliotheken (u.a. Lombok, Guava, Liquibase, Micrometer ...)
- Kommunikation mit REST/OpenAPI und protobuf
- Persistenz mit PostgreSQL

Infrastruktur
- Build mit Maven, Gradle und fastlane
- Docker, Docker Compose (zu Testzwecken), in Produktion auf Kubernetes
- S3 und Content Delivery Network (CDN)

67

Agenda

1 Wozu Architekturüberblicke?

2 Aufgabenbezogene Inhalte

3 Lösungsbezogene Inhalte

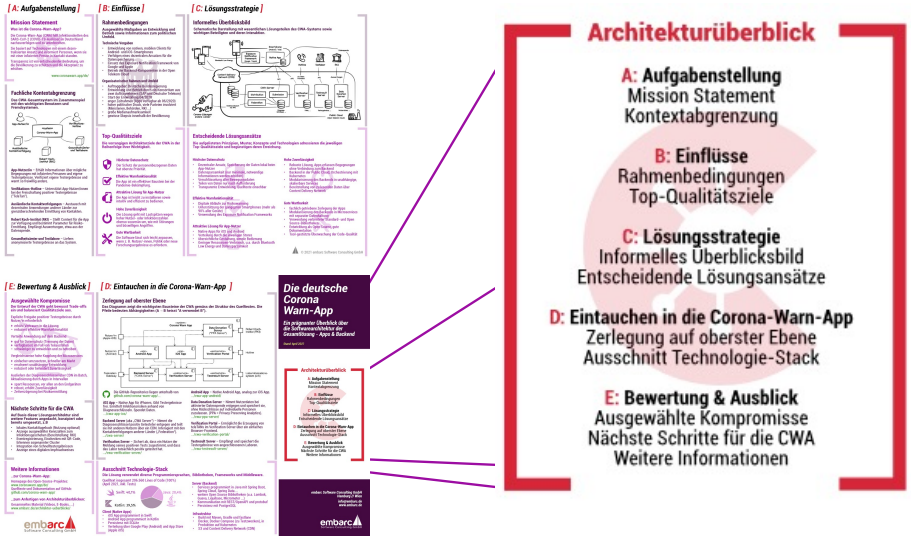
4 Tipps zum Anfertigen Eures Überblicks

5 **Zusammenfassung + weitere Informationen**

5

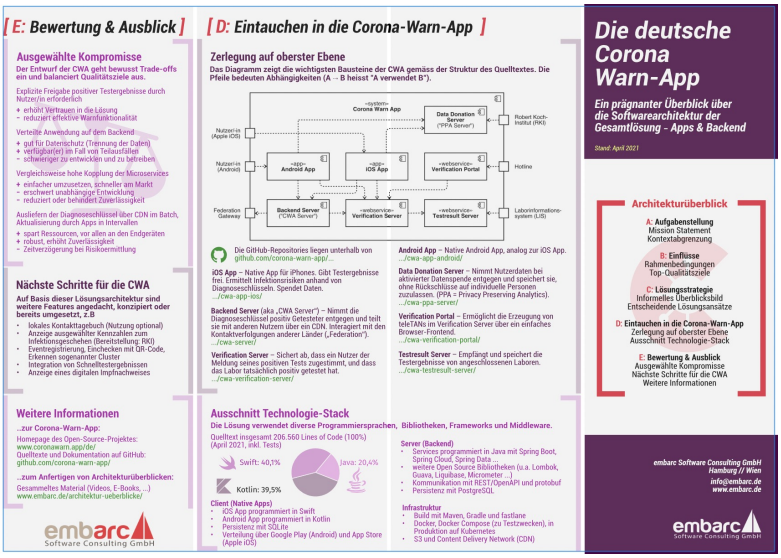
68

Inhalt auf dem Cover



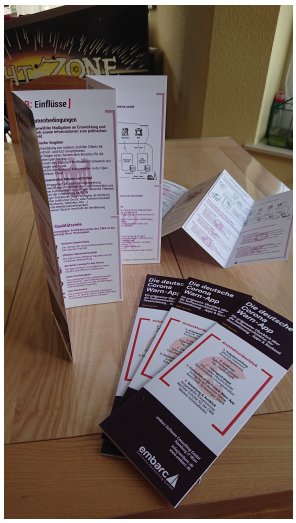
69

Überblicksflyer, 2. Seite komplett



70

Gibt es auch gedruckt!



Wir schicken Euch gerne ein Exemplar zu!
Einfach eine E-Mail senden an

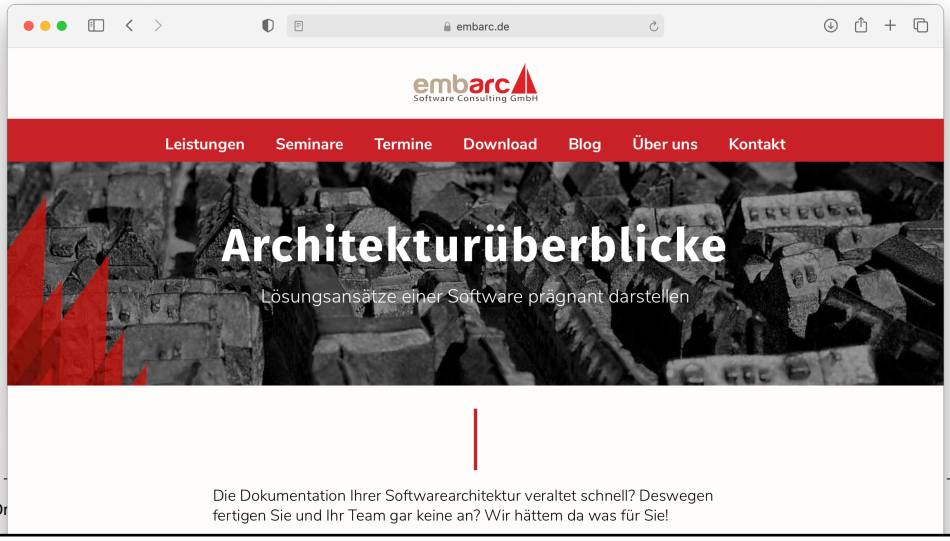
stefan.zoerner@embarc.de

mit Betreff „Flyer“ und Eurer Postadresse im Text, dann geht das los …

71

Und als Download (PDF, zum Selberdrucken ;-)

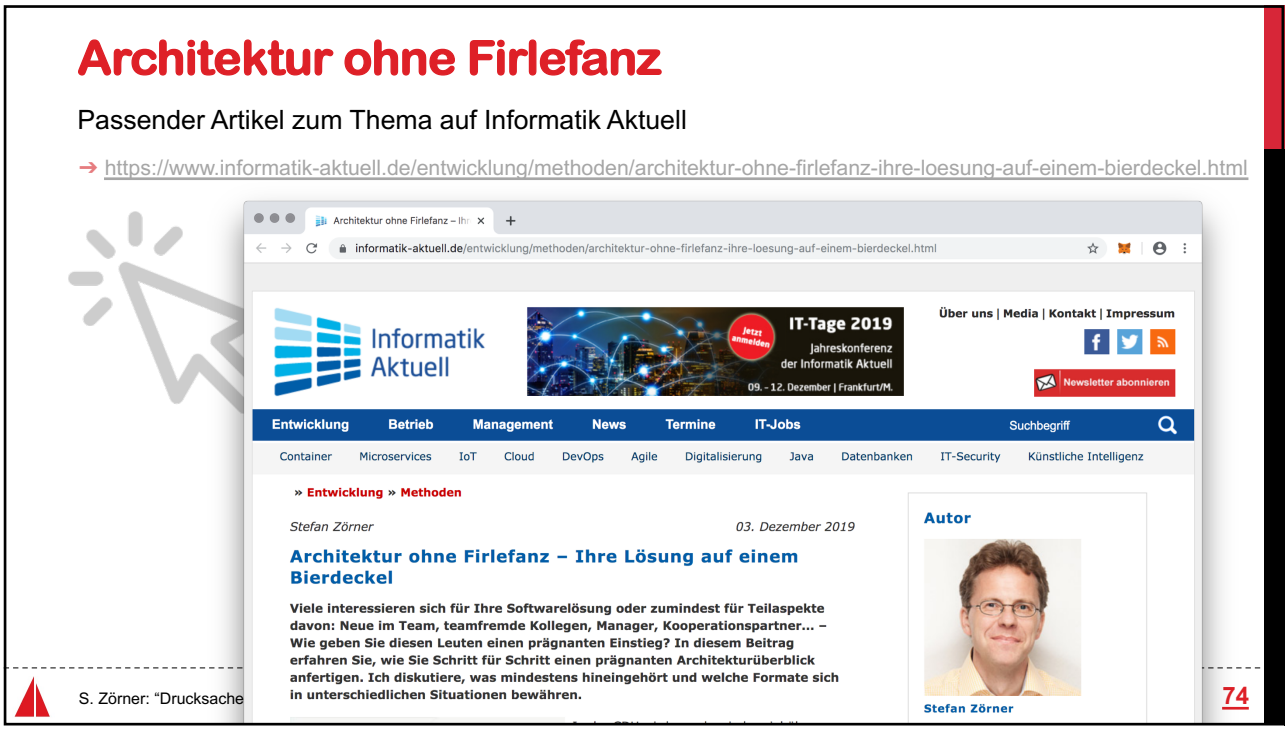
➔ https://www.embarc.de/architektur-ueberblicke/



72

WAS sollen wir bauen?
Was wollen wir erreichen?
Wozu ist es da?
Wem nützt es?
Was soll es tun?
Was braucht es nicht tun?
Woran halten wir uns?
Was soll es exzellent können?
ANFORDERUNGEN
Architekturüberblick
Aufgabe
- Mission Statement
- Kontextabgrenzung
Einflüsse
- Rahmenbedingungen
- Qualitätsziele
Lösungsansätze
- Architekturstil
- Technologie-Stack
- Konzepte
- Prinzipien
- Zerlegung
- ....
Lösungsstrategie (plus Überblicksbild) als Brücke
WIE sieht die Lösung aus?
Wie erreichen wir das?
Welchen Mustern folgen wir?
Was verwenden wir?
Was leitet uns?
Woraus besteht es?
Wie ist es strukturiert?
ENTSCHEIDUNGEN
S. Zörner: "Drucksache. Prägnante Architekturüberblicke"

73


Architektur ohne Firlefanz
Passender Artikel zum Thema auf Informatik Aktuell
https://www.informatik-aktuell.de/entwicklung/methoden/architektur-ohne-firlefanz-ihre-loesung-auf-einem-bierdeckel.html
informatik-aktuell.de/entwicklung/methoden/architektur-ohne-firlefanz-ihre-loesung-auf-einem-bierdeckel.html
Informatik Aktuell
IT-Tage 2019
Jahreskonferenz der Informatik Aktuell
09. - 12. Dezember | Frankfurt/M.
Über uns | Media | Kontakt | Impressum
Newsletter abonnieren
Entwicklung Betrieb Management News Termine IT-Jobs Suchbegriff
Container Microservices IoT Cloud DevOps Agile Digitalisierung Java Datenbanken IT-Security Künstliche Intelligenz
» Entwicklung » Methoden
Stefan Zörner
03. Dezember 2019
Architektur ohne Firlefanz – Ihre Lösung auf einem Bierdeckel
Viele interessieren sich für Ihre Softwarelösung oder zumindest für Teilaspekte davon: Neue im Team, teamfremde Kollegen, Manager, Kooperationspartner... – Wie geben Sie diesen Leuten einen prägnanten Einstieg? In diesem Beitrag erfahren Sie, wie Sie Schritt für Schritt einen prägnanten Architekturüberblick anfertigen. Ich diskutiere, was mindestens hineingehört und welche Formate sich in unterschiedlichen Situationen bewähren.
Autor
Stefan Zörner
S. Zörner: "Drucksache
74

74

Vielen Dank.

Ich freue mich auf Eure Fragen!

sz@embarc.de

@StefanZoerner

➔ xing.to/szr

DOWNLOAD FOLIEN:
https://www.embarc.de/download/

embarc
Software Consulting GmbH

75