HTML5-Anwendung in Cordova für iOS und Android

Benjamin Brachmann, Jan Seewald

Hochschule Osnabrück

Fakultät Ingenieurwissenschaften und Informatik

Barbarastr. 16, D-49076 Osnabrück

benjamin.brachmann@hs-osnabrueck.de

jan.seewald@hs-osnabrueck.de

19. August 2017

**Zusammenfassung**

In diesem Projekt wurde eine HTML5-Anwendung in Cordova für Android- und iOS-Geräte entwickelt, die als Steuergerät für Smart-Objects dient. Sie kann diese Smart-Objects auflisten, auswerten und steuern, sodass eine Fernanalyse und -wartung jener ermöglicht wird.

# **1 Einleitung**

## Esposé

In dieser Projektarbeit soll eine HTML5-Applikation als Erweiterung für eine IoT-Hausarbeit konzipiert und entwickelt werden. In der Applikation sollen im System registrierte Smart-Objects gelistet und anschließend ausgewählt werden können. Nach der Auswahl eines Smart-Objects werden Daten und Messwerte in Form von Diagrammen dargestellt. Ist das Smart-Object eine Kamera, so werden Bilder oder Videos übertragen und dargestellt. Falls das ausgewählte Smart-Object Aktoren besitzt, können diese durch Buttons und andere Oberflächen in der Applikation ferngesteuert werden. Eine „Split”- bzw. „Multi-View" soll die Betrachtung mehrerer Smart-Objects zur selben Zeit ermöglichen. Für die Netzwerkkommunikation werden WebRTC und WebSockets verwendet, die MQTT-over-WebSocket und SIP-over-WebSocket versenden. AngularJs 2 soll als Grundgerüst für die Entwicklung der Applikation dienen. Cordova oder PhoneGap soll als Cross-Plattform Technologie eingesetzt werden, um diese Applikation nativ auf iOS- und Android-Geräten lauffähig zu machen.

## Zielsetzung

Die Aufgabe besteht darin, ein System zur Fernanalyse und -wartung zu konzipieren, mit welchem man nicht-lokale Geräte über ein benutzerfreundliches Interface einsehen, auswerten und steuern kann. Ziel in diesem Projekt ist die Erstellung einer mobilen Anwendung, welche über Nachrichtenprotokolle Daten mit einem Gateway austauscht und die Daten in der Anwendung aufbereitet darstellt und somit als Interface für das System dient.

## Struktur

Im ersten Kapitel wird in das Projekt eingeleitet und ein grober Überblick gegeben. Im darauffolgenden wird über die Kozeption

*Hier sollte angegeben werden:*

* *Problemstellung: Anwendungsfall, auch: Ausschlusskriterien (was wird nicht adressiert?)*
* *Übersicht des Lösungsansatzes*
* *Struktur des Dokumentes*

*Diese Vorlage ist recht einfach gehalten und orientiert sich stilistisch an Vorlagen wie sie für wissenschaftliche Veröffentlichungen im Springer-Verlag (Lecture Notes on Informatics) verwendet.*

*Auf ein Inhaltsverzeichnis kann* ***verzichtet*** *werden.*

- ~~Nachrichtenformat (1,5) + initialisieren (Start)~~  
 - Konzept + Einleitung (Grund / Anwendung) (1,5)  
 - Multiview (0,5)  
 - Fazit (1)  
 - Programme (1)  
 - ~~LanguageService~~ (0,5)  
 - Leistung (X-Plattform) (2,5)  
 - Cordova (1,5)  
 - MQTT.js (0,5)  
 - Messaging zwischen Backend und View (1)  
 - Vue.js (1,5)  
 - Bulma (SASS/SCSS) (0,5)  
 - Kameraansteuerung (1)  
 - Bilder (3,5)  
 - Diagramme (3)  
  
 = (22,5)

# 

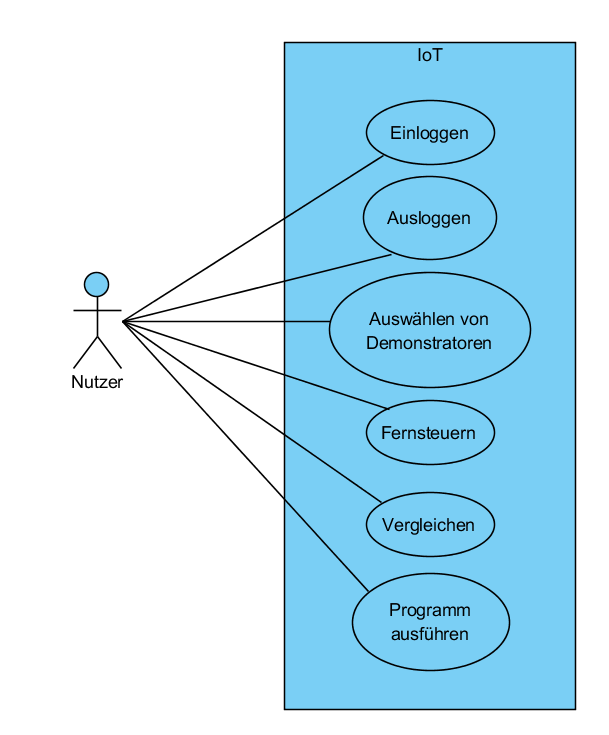
# **2 Konzept**

*Hier sollte der Lösungsansatz beschrieben werden.*

*Sie werden in Ihrer Arbeit sicherlich auf Quellen zurückgreifen, die entsprechend referenziert werden müssen. Referenzen auf Einträge im Literaturverzeichnis können nach der Harvard-Methode vorgenommen werden. Zum Beispiel: diese Vorlage basiert in Teilen auf [ABC01], die wiederum auf Standards in [AB00] zurückgreift. Versuchen Sie auch Monographien zu finden (nicht nur Internet-Referenzen)!*

*Unterkapitel verbessern die Strukturierung der Arbeit.*

## **2.1 Unterkapitel**



## 

## 

## **2.1 Nachrichtenformat**

Das Nachrichtenformat ist ein relevanter Aspekt in der Übertragung der Daten über MQTT. Es muss gewährleistet werden, dass die Nachrichten in jeweiliger Kategorie ein bestimmtes Format haben, sodass die Geräte keinen großen Aufwand zur Entschlüsselung haben. Da die Nachrichten in MQTT datenagnostisch sein, also jegliches Format haben können, gibt es keine Einschränkungen.

Es wurde letztendlich auf das Datenformat JSON gesetzt, welches bereits eine klare Syntax bietet und sich leicht implementieren lässt, gerade in JavaScript.

Um die Kategorien der Nachrichten zu trennen, werden für jene unterschiedliche Topics auf dem MQTT-Broker verwendet.

Mithilfe von Topics lassen sich hierarchische Strukturen aufbauen, unter denen sich die Nachrichten anordnen können. Damit jedes Smart-Object ein einzigartiges Topic bekommt, bekommt jede Instanz wie Server, Gateway und das Smart-Object eine ID, um daraus ein Topic bilden zu können. Ein Topic könnte somit so aussehen:

S1/G2/**SM4**

Das Smart-Object Nr. 4 befindet sich an dem Gateway 2, welches wiederum mit dem Server 1 kommuniziert. So kann direkt eingesehen werden, wo sich das Smart-Object befindet, da das Gateway den Ort preisgibt. Topic für das Gateway 2 am Server 1:

S1/G2

Daten in folgender Form:

{

name: ‚Gateway Nummer 2‘,

location: ‚Osnabrueck‘

}

In unserem System verwenden wir 4 verschiedene Kategorien für unsere Smart-Objects, in denen Nachrichten ausgetauscht werden. Diese wären:

* Informationen

Hier schreibt das Smart-Object relevante Daten wie Name, Typ oder Status rein.

Dabei wird in das Topic I0 veröffentlicht, um die Notation beizubehalten.

Das Smart-Object kann bereit, beschäftigt, reserviert und offline sein.

* Topic: S1/G1/SM1/**I0**
  + {

name: ‚SM-One‘,

description: ‚BlaBla‘,

status: ‚READY‘ (‚READY‘, ‚BUSY‘ ‚RESERVED‘, ‚OFFLINE‘),

type: ‚CAMERA‘ (‚CAMERA‘, ‚SMARTOBJECT‘)

}

* Funktionen

Hier werden alle Funktionen aufgelistet, die das Smart-Object ausführen kann

* Topic: S1/G1/SM1/**F1**
  + {

name: ‚Move in direction: right‘,

parameter\_type: ‚INTEGER‘, (‚DOUBLE‘)

min: ‚0‘

max: ‚8‘

}

* Topic: S1/G1/SM1/**F2**
  + {

name: ‚Display Message‘,

parameter\_type: ‚STRING‘,

min: ‚0‘

max: ‚10‘

}

* Topic: S1/G1/SM1/**F3**
  + {

name: ‚Toggle Light‘,

parameter\_type: ‚BOOLEAN‘

}

* Topic: S1/G1/SM1/**F4**
  + {

name: ‚Drive Motor X forward‘,

parameter\_type: ‚NONE‘

}

* Daten

Hier werden alle Daten von Sensoren aufgelistet und regelmäßig aktualisiert

* Topic: S1/G1/SM1/**D0**
  + {

name: ‚Light Brightness‘, /S1/G1/SM1/D/1/

parameter\_type: ‚DOUBLE‘

min: ‚0.0‘

max: ‚1.0‘

value: ‚0.1122‘

}

* Topic: S1/G1/SM1/**D1**
  + {

name: ‚Door open‘, /S1/G1/SM1/D/2/

parameter\_type: ‚BOOLEAN‘

value: ‚0‘

}

* Aktionen

Das Sub-Topic A0 wird vom Smart-Object abonniert und die Steuerungsgeräte

veröffentlichen dort die auszuführenden Funktionen.

* Topic: S1/G1/SM1/**A0**
  + {

functionName: ‚F1‘

parameters: ‚A cool String‚

}

Zusätzlich gibt es noch die Funktion, die jedes Smart-Object haben muss. Sie trägt immer das Sub-Topic **F0** und gewährleistet einen Notausschalter, der das Gerät neustartet und somit alle Aktoren stoppt.

### Initialisieren

Beim Hochfahren des Smart-Objects werden alle diese Topics initial veröffentlicht, sodass das Gateway und auch die App die nötigen Informationen weitergeben bzw. darstellen können. In der Prototypenphase hat das Smart-Object zusätzlich noch ein Topic abonniert, in welchem sich die App bei Start meldet und daraufhin alle Informationen aus den o.g. Topics ebenfalls initial erhält, sodass aktuell kein Gateway zur Übermittlung der Informationen notwendig ist. In einer späteren Phase soll dies das Gateway erledigen, bei dem sich alle Smart-Objects und mobilen Endgeräte registrieren werden.

## 

## 

## **2.1 Language Service**

Die mobile Anwendung besitzt außerdem die Möglichkeit, die Sprache zu ändern. Für den Prototypen werden vorerst nur Englisch und Deutsch als auswählbare Sprachen verwendet. Die wichtigsten Begriffe in der App wurden durch Variablen ersetzt und dank Vue.js mit ein paar einfachen Befehlen und einer JSON-Datei mit korrekten Zuordnungen der jeweiligen Begriffe in der Endansicht ausgetauscht. Die entsprechenden Komponenten laden sich dann neu, um die geänderte Sprache korrekt anzuzeigen.

[Screenshot?]

## **2.1 Multiview**

Die Multiview ermöglicht es, mehrere Smart Objects auf einer Ansicht anzuzeigen. So kann beispielsweise eine Kamera neben einem Industrie-Demonstrator dargestellt werden, welche einen Livebild vom Demonstrator übersendet. So können die gestarteten Aktionen eingesehen werden und gibt dem Nutzer die Sicherheit, keine Fehler zu machen.

## **2.1 Language Service**

## **2.1 Language Service**

**3 Realisierung**

**Featureliste**

* **Login**
* **Auflistung der Smart-Objects**
  + JSON-Objekt als Init
  + Durch anklicken Vorschau, aufklappen der Informationen
  + Durch auswählen des Smart-Objects, zur nächsten Seite  
    (Fernsteuerung, Fernanalyse)
* **Smart-Object**
  + Besitzt:
    - Aktoren
      * Lassen sich Fernsteuern
    - Sensoren
      * Lassen sich auslesen
  + SO1
    - Demonstrator
  + SO2
    - Kamera
* IoT-Smart-Object
  + Kennt seine Fähigkeiten
  + Bietet diese und weitere Daten nach außen an
  + Diese Fähigkeiten können aus der Ferne ausgeführt werden
  + MQTT
  + Log-Push (OK, ERROR)
  + Willkommens-nachricht (Das kann ich, der bin ich)
  + Konfigurationsdatei
  + Command-Basierend
* Gateway
  + Queued Befehle
  + checked ob noch da
  + Leitet weiter
* Server (BL)
  + MQTT Broker
* Server (FS)
  + Wie immer
* Mobiles Endgerät

# 

# 

# **4 Test**

Was und wie wurde getestet? Tests können anhand von Anwendungsfällen entwickelt werden.

# **5 Fazit & Ausblick**

*Ein Fazit zur Arbeit:*

* *Was ist gelungen? Was ist nicht gelungen?*
* *Welche Probleme sind aufgetaucht / gelöst worden?*

*Vergleichen Sie Ihre Lösung ggf. mit anderen bekannten Ansätzen.*

*Geben Sie auch einen Ausblick: wie kann das Konzept noch erweitert werden? Welche alternativen Lösungen sind noch denkbar?*

Aufgrund der

# **Literatur**

[AB00] Abel, K.; Bibel, U: Formatierungsrichtlinien für Tagungsbände. Format-Verlag, Bonn, 2000.

[ABC01] Abraham, N.; Bibel, U.; Corleone, P.: Formatting Contributions for LNI. In (Glück, H.I. Hrsg.): Proc. 7th Int. Conf. on Formatting of Workshop-Proceedings, New York 1999. Noah & Sons, San Francisco, 2001; S. 46-53.

[Az99] Azubi, L. et.al.: Die Fußnote in LNI-Bänden. In (Glück, H.I.; Gans, G., Hrsg.): Formatierung leicht gemacht – eine Einführung. Format-Verlag, Bonn, 1999; S. 135-162

[Ez99] Ezgarani, O.: The Magic Format – Your Way to Pretty Books, Noah & Sons, 2000.

# **Erklärung**

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne Verwendung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Benjamin Brachmann) (Jan Seewald)

Osnabrück, den 19. August 2017 Osnabrück, den 19. August 2017

[1] Dies ist eine Fußnote.

[Konzeption und Entwicklung einer mobilen Anwendung (= Entwicklungsprojekt): Anwendung sollte nicht-lokal sein / Serverkommunikation beinhalten, bei mehr als einem Teilnehmer sollten mehrere Plattformen bedient werden, die Konzeptionierung sollte signifikante Herausforderungen beinhalten)]

[Bearbeitungshinweise:

Konzept & Lösung in den Vordergrund stellen: Problem erläutern und eigene Lösung motivieren.]

Literaturverzeichnis: auch Monographien, nicht nur Links zu Tools etc.

Zum Ende der Ausarbeitung: ​Herangehensweise herausstellen: Sollte strukturiert sein.]

**Analyse**

[Recherche durchführen: Welche Ansätze gibt es schon?]

Fazit

[Bewertung / Evaluation wichtig: Auch negative Ergebnisse dokumentieren]