

# POSITION PREDICTION BASED ON MOVEMENT DATA

## SYSTEM DESIGN DOCUMENT

*Authors:*

Timo Jockers, Oliver Mänder, Benjamin Moser,  
Manuel Prinz, Sebastian Strumbelj, Simon Suckut

# Contents

<b>1</b>	<b>Änderungen</b>	<b>2</b>
<b>2</b>	<b>Einleitung</b>	<b>2</b>
2.1	Zweck . . . . .	2
2.2	Design-Ziele . . . . .	2
2.3	Definitionen und Abkürzungen . . . . .	3
2.4	Referenzen . . . . .	4
2.5	Überblick . . . . .	4
<b>3</b>	<b>Aktuelle Architektur</b>	<b>4</b>
<b>4</b>	<b>Vorgeschlagene Architektur</b>	<b>4</b>
4.1	Teilsysteme . . . . .	4
4.2	Hardware-Software-Zuordnung . . . . .	4
4.3	Management von persistenten Daten . . . . .	5
4.4	Sicherheit . . . . .	5
4.5	Global Software Control . . . . .	6
4.5.1	Interfaces . . . . .	7
4.6	Boundary Conditions . . . . .	7
<b>5</b>	<b>Anhänge</b>	<b>7</b>
5.1	Geänderte Anforderungen . . . . .	7

# 1 Änderungen

Version	Author	Description of changes	Date
1.0	SS	Aufbau der App in Android Studio	x.5.2018

Legend:

SSJ = Sebastian Strumbelj

BM = Benjamin Moser

MP = Manuel Prinz

OM = Oliver Mänder

SS = Simon Suckut

TJ = Timo Jockers

ALL = SSJ, BM, MP, OM, SS, TJ

## 2 Einleitung

### 2.1 Zweck

Das Erstellen einer App, um einem Ornithologen das Finden eines Vogels zu erleichtern, indem sie dessen zukünftige Position mit einem Vorhersagemodell abschätzt.

### 2.2 Design-Ziele

Diese Dokumentation spezifiziert alle Eigenschaften, die unsere App benötigt, um deren Zweck zu Erfüllen. Um eine Vorhersage für Vögel zu treffen und dem Ornithologen bei seiner Suche zu unterstützen, sollte die App folgende Funktionen beinhalten:

- Aufbereiten und Speichern der vorhandenen und benötigten Daten
- Vorhersageberechnung

- Visualisierung der Daten
- Userinterface zur Interaktion

Die App beinhaltet keine Funktionen die folgenden entsprechen:

- Daten aus der Vergangenheit aufwendig visualisieren, um das vergangene Verhalten der Vögel zu studieren
- **TODO?**

## 2.3 Definitionen und Abkürzungen

**App, Applikation** Soweit nicht anders angemerkt wird hiermit die zu entwickelnde Android-Applikation bezeichnet.

**Forscher** Personen, die wissenschaftlich das Verhalten von Vögeln untersuchen.

**Mobile Daten** Ein Smartphone kann sich entweder über ein WiFi-Netzwerk oder über ein mobiles Datennetzwerk (3G, LTE, ...) mit dem Internet verbinden.

**Vorhersage-Modell/Algorithmus** Die zur Berechnung einer Vorhersage verwendete Methodik.

**API** *Application Programming Interface*, hier insbesondere Schnittstellen mit Providern wie *Movebank* oder *Cesium*.

**Outdoor-Modus, Normal-Modus** Die Applikation verfügt über zwei verschiedene Funktionsmodi. Einer davon ist der sog. Outdoor-Modus.

**Cesium** Ist eine Javascript-Bibliothek für die Arbeit mit dreidimensionalen Kartendaten.

**Offline-Karten** Ist ein 2D-Karten-Provider der Kartendaten liefert, wenn keine Internetverbindung besteht.

**SRS** Software Requirements Document

## 2.4 Referenzen

- SRS
- Treffen mit den Betreuern
- Treffen mit einem Forscher
- Notizen zu Recherche über Nutzungsbedingungen und Lizenzen von Systemkomponenten.

## 2.5 Überblick

Der erste Teil legt grundlegende Ideen und Definitionen der App fest. Der zweite Teil beschäftigt sich mit bereits existierenden und ähnlichen Systemen. Der dritte Teil liefert eine Beschreibung für die für unsere App genutzte Architektur.

# 3 Aktuelle Architektur

”Animal Tracker” ist bereits eine App, welche Vogeldaten visualisiert. Allerdings berechnet diese keine Vorhersagen für die Zukunft und fokussiert sich dementsprechend nur auf die Darstellung vergangener Daten. Interne Architekturen dieser App sind allerdings nur gering bekannt.

# 4 Vorgeschlagene Architektur

## 4.1 Teilsysteme

Timo

## 4.2 Hardware-Software-Zuordnung

Timo

### 4.3 Management von persistenten Daten

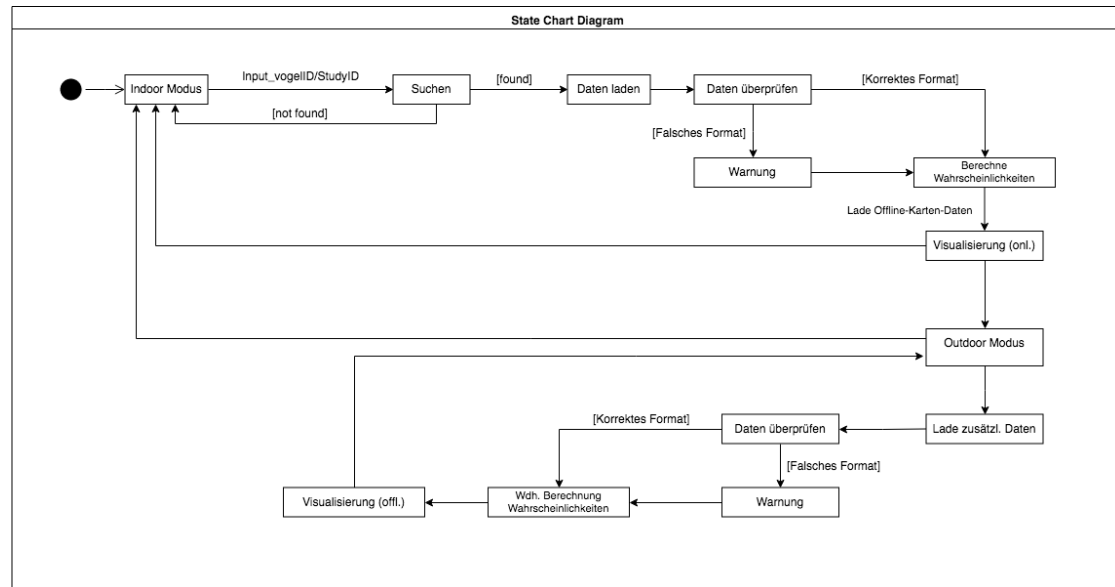
Um Daten nach dem Schließen der Applikation zu behalten, die Menge des Datenvolumens für Anfragen an die Movebank zu reduzieren und um Daten leichter verwalten zu können verfügt die Applikation über eine Schnittstelle zum Parsen und Speichern von XML-Dateien und eine SQLite-Datenbank. Außerdem werden Karten für den Offline-Modus lokal gespeichert.

- *SQLite-Datenbank*: In dieser Datenbank werden Daten von der Movebank lokal zwischen gespeichert um die Menge an Anfragen an die Movebank so gering wie möglich zu halten, die Menge des verbrauchten Datenvolumens zu reduzieren und die Daten leichter verwalten zu können.
- *XML-Parser*: In der Applikation getätigte Einstellungen werden persistent in einer XML-Datei gespeichert.
- *Offline-Karten*: Karten-Daten werden in dem vom Kartenanbieter verwendeten Format lokal zwischengespeichert.

### 4.4 Sicherheit

Mit einem Benutzernamen und Passwort kann der Zugriff für die Movebank-Daten erweitert werden. Es gibt aber keinen weiteren Zugriff (z.B. auf die Verwaltung) der App.

## 4.5 Global Software Control



Das Global-Software-Control-Diagramm beschreibt das allgemeine Verhalten der App. Es besteht aus dem allgemeinen Ablauf der App im Indoor- (obere Hälfte) und Outdoormodus (untere Hälfte).

Zu Beginn befinden wir uns im Indoor-Modus und können nun mit der VogelID und StudyID nach einem Vogel suchen. Ist der Vogel nicht verfügbar, so haben wir keine Möglichkeit weiter zu machen und beginnen von vorne. Ist er vorhanden, so laden wir dessen Daten. Anschließend werden diese geprüft bzw. geparkt. Im Fall, dass die Daten formal falsch sind (siehe SRS für die Definition von formal falsch), werden diese trotzdem zur Berechnung benutzt. Allerdings wird eine Warnung über die Qualität der Daten erscheinen. Nach dem Berechnen der Wahrscheinlichkeiten, werden die Kartendaten und weitere Daten für die Visualisierung heruntergeladen, um diese auch ohne Internetverbindung visualisieren zu können. Solange wir uns aber im Indoor-Modus befinden, können diese mit Cesium visualisiert werden. Sollen andere Vögel untersucht werden, so kann die Suche von vorne beginnen.

Geht der Forscher nun in das Feld, schaltet dieser in den Outdoor-Modus. Der Ablauf ist nun ähnlich wie im Indoor-Modus. Es können zusätzliche oder neue Daten des Vogels über die mobilen Daten geladen werden. Sie fließen in die erneute Wahrscheinlichkeitsberechnung ein und aktualisieren die Visualisierung, die nun allerdings mit einem Offline-Karten-Provider dargestellt wird.

#### 4.5.1 Interfaces

### 4.6 Boundary Conditions

Oliver

## 5 Anhänge

### 5.1 Geänderte Anforderungen